3/17

Principes en doelstellingen van vage logica

Inhoud

3/17.1

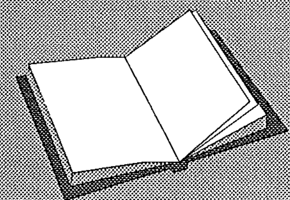
Het principe van vage logica (fuzzy logic)

Inleiding

Het is niet altijd ja of nee
De normale logische schakelingen werken alleen met gegevens, die worden geklassificeerd in twee groepen of verzamelingen: "waar" en "niet waar"; "hoog" en "laag", "ja" en "nee". Dat met dergelijke "harde" logica de meest ingewikkelde problemen kunnen worden opgelost, bewijst wel het feit dat er computers bestaan. Dergelijke apparaten zijn immers typische voorbeelden van de toepassing van "harde" logica.
Toch zijn er problemen die maar moeilijk te omschrijven zijn met de keiharde begrippen "ja" en "nee". Vraag tien verschillende bioscoopbezoekers naar hun waardering van de film en de antwoorden zullen uit elkaar lopen van "slaapverwekkend" over "gaat wel" en "niet slecht" tot "geweldig". Vraag dezelfde tien bezoekers hun waardering alleen uit te drukken door met "ja" of "nee" te antwoorden op de vraag "vindt u deze film de moeite waard?" en er zal een zeer vertekend beeld van de waardering ontstaan.

Meet- en regelelektronica
Met de toenemende digitalisering van de elektronica wordt men steeds vaker geconfronteerd met het gegeven dat bepaalde processen niet zo gemakkelijk te vangen zijn in harde logica. Ingewikkelde regelsystemen, die vroeger met behulp van analoge schakelingen (de zogenoemde analoge computers) werden uitgevoerd, worden steeds vaker volledig digitaal geregeld met computers. Maar die computers eisen wél keiharde logica, zowel in hun invoergegevens als in hun beslissingen! Deze beslissingsregels, waarmee de computer uit de invoergegevens bepaalde uitvoergegevens afleidt, moeten volledig en ondubbelzinnig bepaald zijn. Met de beslissingsregel "als het glas in de oven iets vloeibaarder wordt dan uit ervaring wenselijk is, dan moet de gaskraan ietsjes dicht gedraaid worden" kan een computer helemaal niets beginnen. Natuurlijk zou een dergelijk probleem nog wel omgezet kunnen worden in harde logica door de temperatuur van het glas als maatstaf te nemen voor de vloeibaarheid ervan, de temperatuur analoog te meten, om te zetten in harde logische

LEES OOK:
Hoofdstuk 3/6.4
Hoofdstuk 3/6.5
Hoofdstuk 3/17.2
Hoofdstuk 3/17.3



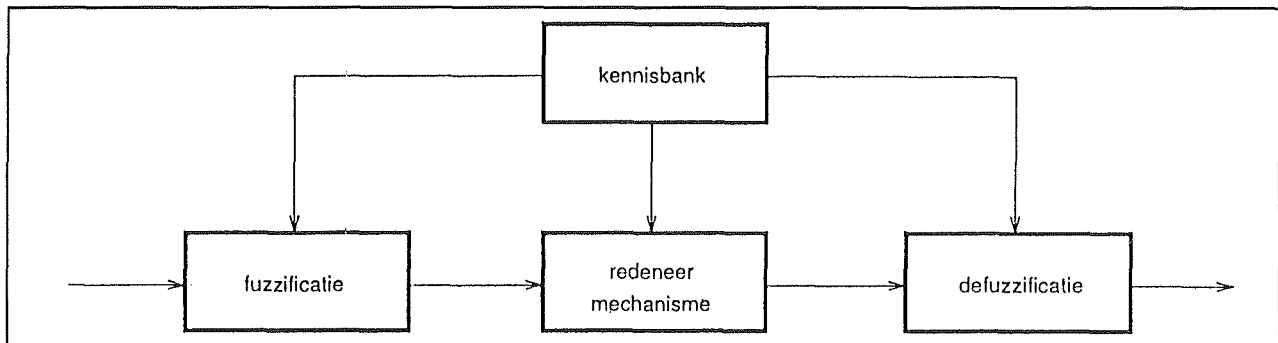
17.1 Het principe van vage logica (fuzzy logic)

digitale codes en deze door een aantal harde beslissingsregels te laten evalueren. Dergelijke Boolse vergelijkingen leveren dan weer harde digitale uitgangscodes op, die weer omgezet kunnen worden in analoge signalen die gaskranen dicht kunnen draaien.
Maar bij zeer ingewikkelde industriële regelingen, waar tientallen ingangsgrootheden invloed hebben op een proces en bovendien een belangrijke factor, namelijk:
de ervaring van de menselijke procesbegeleider
heel veel invloed heeft op het eindresultaat, komt men snel in de problemen.

Vage logica
Vandaar dat in de jaren zeventig een geheel nieuwe benadering van dergelijke problemen werd gezocht. Alle elementen van een verzameling (bijvoorbeeld alle oordelen van de bioscooopbezoekers) worden nu niet in twee keiharde set's ingedeeld ("ja" en "nee") maar volgens een glijdende schaal van "0 % ja" (en dus "100 % nee" tot "100 % ja" (en dus "0 % nee"). Een dergelijke benadering sluit veel beter aan bij de manier waarop mensen informatie classificeren.
Nu mag men niet de vergissing begaan om te denken dat in- en uitvoergegevens niet meer feitelijk "hard" zijn! Er is weinig vaags aan de vage logica! De kunst van het principe is de in- en uitgangsgegevens te vangen in zogenoemde "lidmaatschapsfuncties" en nadien beslissingsregels op te stellen waarmee het systeem uit de voeten kan.

Lotfi Zadeh
De fundamenten van de vage logica zijn, zoals tegenwoordig met zovele technieken het geval is, terug te vinden in de theoretische wiskunde. In 1965 publiceerde de wiskundige L. A. Zadeh van de Berkeley universiteit in de Verenigde Staten een boek, getiteld "Fuzzy Sets in Information and Control". De bedoeling was een wiskundige theorie op te stellen, waarmee taalkundige stellingen vertaald konden worden naar wiskundige formules. Het was Zadeh onmiddellijk duidelijk dat dit alleen maar kon als werd afgeweken van de klassieke verzamelingsleer, waarbij een element ondubbelzinnig tot een set behoort. Vandaar voerde hij het begrip "lidmaatschapsgraad" in, waarmee wordt aangegeven in hoeverre een bepaald element van de verzameling tot een bepaalde set behoort. Deze lidmaatschapsgraad kan dan worden omgezet in een lidmaatschapsfunctie, afgekort tot LF, in wezen niets anders dan een grafische voorstelling van de graad van lidmaatschap van ieder element uit de verzameling bij een bepaalde set. Op deze manier worden alle ingangsvariabelen omgezet in een of meerdere LF's. Dit proces noemt men de "fuzzificatie". Ook de uitgangsvariabelen moeten op een identieke manier gefuzzificeerd worden.
Nadien moeten bepaalde relaties tussen deze in- en uitgangsgrootheden worden opgesteld. Dit noemt men het "redeneer mechanisme". De regels voor dit redeneer mechanisme worden ontleend uit de kennis die de bedenker van het systeem heeft. Vandaar dat vaak wordt gerefereerd naar de "kennisbank" en de regels "ervaringsregels" worden genoemd. Dit proces gebeurt in regels, die welbekend zijn uit de "harde" logica. De meest algemene structuur van een dergelijke regel is:
ALS
aan bepaalde voorwaarden voldaan wordt
DAN
moet op een bepaalde manier een actie ondernomen worden

17.1 Het principe van vage logica (fuzzy logic)



Figuur 3/17.1-1: De drie voornaamste stappen van een fuzzy-proces samengevat.

Het enige verschil is dat de voorwaarden en de acties niet alleen "ja" en "nee" kunnen zijn, maar ook tussenwaarden kunnen hebben.
De laatste stap noemt men de "defuzzificatie". Hierin worden de LF's van de ingangsgrootheden en van de uitgangsgrootheden gekoppeld aan de ervaringsregels, met als gevolg dat er bepaalde uitgangsfuncties ontstaan.

Het fuzzy-proces

Het volledige proces van fuzzy logic kan dus voorgesteld worden zoals samengevat in figuur 3/17.1-1:

- Fuzzificatie:
  Ingangsgrootheden en uitgangsgrootheden worden verwerkt tot lidmaatschapsfuncties, waarbij rekening wordt gehouden met ervaringsregels die in de kennisbank aanwezig zijn.
- Redeneer mechanisme:
  Hierbij is het de bedoeling een aantal ALS... DAN regels op te stellen waarbij de vage ingangsvariabelen gekoppeld worden aan een of meerdere vage uitgangsvariabelen. Deze regels beschrijven het systeem waarop het fuzzy-proces wordt toegepast.
- Defuzzifucatie:
  Uit de regels worden waarden voor de uitgangsvariabelen afgeleid, die niet langer vaag zijn maar concreet, zodat elektronische schakelingen er weer iets mee kunnen aanvangen.

Vage logica in de praktijk

Het zijn vooral Japanse bedrijven geweest die de wiskundige theorie van Zadeh heel snel hebben omgezet in praktische toepassingen. Ontelbaar zijn de Japanse consumentenproducten, zoals videocamera's, wasmachines en zelfs scheerapparaten, die via fuzzy logic worden bestuurd. Een bedrijf als Mitsushita produceerde in het jaar 1992 voor meer dan een miljard dollar producten, die op de een of andere manier het etiket "fuzzy" verdienen.
Ondertussen hebben echter ook Amerikaanse en Europese fabrikanten de waarde van de vage logica ontdekt. Alle vooraanstaande elektronische concerns brengen op dit moment fuzzy-controllers in de handel, waarmee industriële processen geregeld kunnen worden. Naast hardware-oplossingen worden steeds meer software-emulatoren aangeboden, waarmee men fuzzy systemen kan ontwerpen.

De toekomst

Alles lijkt er op te wijzen, dat fuzzy logic een steeds belangrijker rol gaat spelen in de automatisering. Men voorspelt zelfs dat de verdere ontwikkeling van de vage logica tot gevolg zal hebben dat producten niet alleen worden beoordeeld op prijs en

kwaliteit, maar ook op hun "Machine Intelligence Quotient". Deze "MIQ" is een getal dat aangeeft hoe gebruiksvriendelijk en intelligent een bepaald apparaat dank zij vage logica wordt. Camera's waar de gebruiker niets hoeft in te stellen en iedere druk op de knop een bruikbare foto oplevert zijn er al en dergelijke apparaten verdienen een zeer hoge MIQ!

Specifieke toepassingsterreinen
Vage logica zal vooral worden toegepast in die processen, die met "harde" logica heel moeilijk te omschrijven zijn. Een typisch voorbeeld van een dergelijk proces is patroonherkenning. Patroonherkenning staat in het middelpunt van de belangstelling, omdat technieken die in staat zijn met een aan zekerheid grenzende waarschijnlijkheid een bepaald patroon te herkennen een zeer grote toekomst hebben. Patroonherkenning vormt immers het kloppend hart van zeer uiteenlopende praktisch toepassingen zoals:
- elektronische handschriftherkenning;
- elektronische spraakherkenning;
- verminkte gegevens terugwinnen uit beschadigde of gestoorde datastromen;
- automatische sorteersystemen, zoals het sorteren van tomaten op grootte, vorm of rijpheid;
- automatische visuele inspectie door middel van een video-camera van de producten op een lopende band.

Al deze toepassingen hebben een belangrijke eigenschap gemeen en dat is dat de ingangsvariabelen niet exact omschreven kunnen worden. Het heeft geen zin om te proberen vorm en kleur van een tomaat wiskundig te beschrijven in exacte formules. Ieder tomaat zal immers in min of meerdere mate van een dergelijke absolute beschrijving afwijken! Hier kunnen alleen de vage beslissingen van de vage logica bruikbare oplossingen bieden!

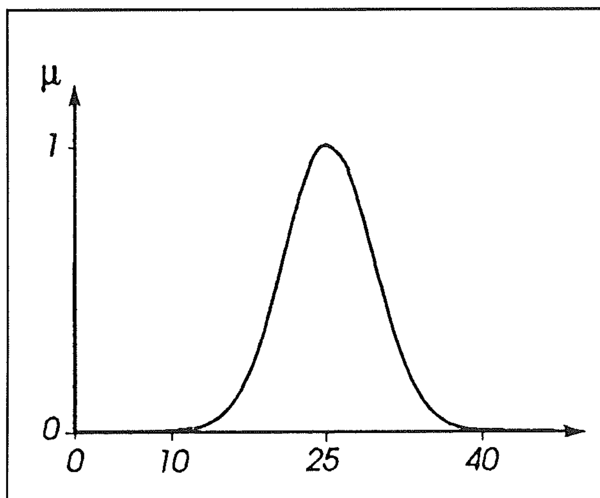
De fuzzificatie

Het principe van de lidmaatschapsfunctie
Zoals reeds gesteld is het voornaamste instrument van de vage logica de "lidmaatschapsfunctie" LF. Omdat de meeste software voor het opstellen van vage oplossingen in het Engels of het Duits is geschreven, is het verstandig de speciale terminologie van de vage logica ook in deze talen te vermelden:
- Engels: membership function MF;
- Duits: Zugehörigkeitsfunktion ZK.

Soms wordt in het Nederlands ook de term "toebehorensfunctie" gebruikt. Het woord lidmaatschap dekt echter veel beter de lading en vandaar dat deze in dit hoofdstuk gebruikt zal worden.
De lidmaatschapsfunctie is een grafiek die aangeeft in hoeverre een bepaald element uit een verzameling tot een bepaalde set behoort. Een voorbeeldje zal dit verduidelijken. Stel dat men aan duizend personen de vraag stelt aan welke leeftijd men denkt als men het begrip "een jonge man" in gedachten neemt. Waarschijnlijk zal er één leeftijd het vaakst genoemd worden, bijvoorbeeld "25 jaar". Rond deze vaakst genoemde leeftijd zullen antwoorden liggen die iets minder vaak genoemd worden, zoals "24 jaar" en "26 jaar". Tot slot zullen er ook antwoorden zijn, die maar door een paar mensen genoemd worden, zoals "40 jaar" en "10 jaar".
Men kan nu aan de meest genoemde leeftijd een waarde 1 toekennen en alle andere leeftijden hieraan relateren. Als het

17.1 Het principe van vage logica (fuzzy logic)

antwoord "25 jaar" 120 keer genoemd wordt en het antwoord "15 jaar" 12 keer, dan relateert men het antwoord "15 jaar" aan het antwoord "25 jaar" door er de waarde 0,1 aan toe te kennen. Dit antwoord wordt ten opzichte van het antwoord "25 jaar" immers slechts één tiende aantal keren genoemd.



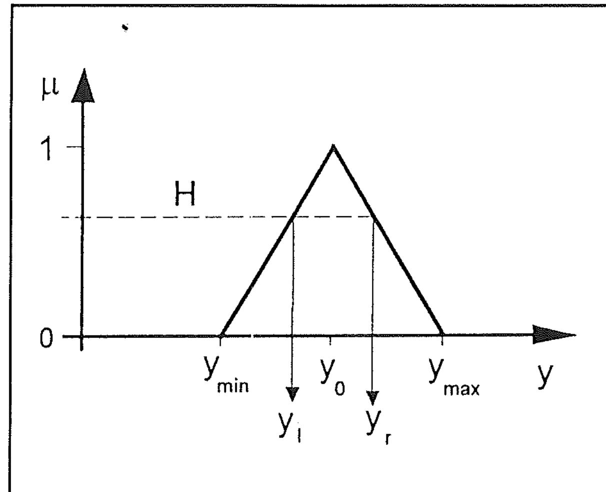
Figuur 3/17.1-2: De als voorbeeld opgestelde lidmaatschapsfunctie voor het begrip "een jonge man".

Men is nu klaar voor het opstellen van de lidmaatschapsfunctie voor het begrip "een jonge man". Die functie bestaat uit de grafiek die getekend is in figuur 3/17.1-2. Op de vertikale as worden alle aan het antwoord "25 jaar" gerelateerde getallen uitgezet, op de horizontale as alle verkregen antwoorden. Lezers die iets van wiskunde afweten zullen in de getekende grafiek onmiddellijk de beroemde Gaussiaanse verdelingscurve herkennen.
Een lidmaatschapsfunctie heeft dus steeds een vertikale as die van 0 tot 1 loopt. De indeling en schaal van de horizontale as is natuurlijk afhankelijk van het verschijnsel dat men wil fuzzificeren.

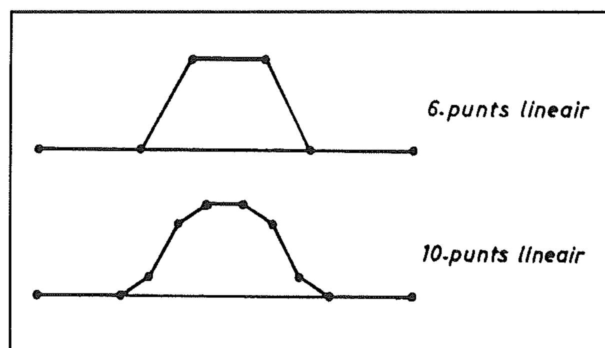
Nog wat nieuwe begrippen
Aan de hand van de definitie van een lidmaatschapsfunctie kunnen nog enige nieuwe begrippen gedefinieerd worden.
Wat in figuur 3/17.1-2 gebeurt is, is dat een vaag begrip als "een jonge man" *gequantificeerd* werd. Er worden waarden aan toegekend, waardoor het voor een marsmannetje mogelijk zou zijn een relatie te leggen tussen leeftijden van aardbewoners en het begrip "een jonge man", zelfs zonder dat hij ooit een aardbewoner had gezien.
Aan de hand van de lidmaatschapsfunctie kan men aan iedere leeftijd een zogenoemde *lidmaatschapsgraad* toekennen. Deze graad wordt standaard voorgesteld door de Griekse letter μ, uit de spreken als mhu. In het voorbeeld is de lidmaatschapsgraad van het antwoord "25 jaar" gelijk aan 1. Het zal duidelijk zijn dat de waarde van μ steeds begrensd wordt door het minimum 0 en het maximum 1.
Het volledig bereik van de lidmaatschapsfunctie noemt men de *fuzzy-set.* Figuur 3/17.1-2 geeft dus de fuzzy-set voor het vage begrip "een jonge man".
Het vage begrip, waarvoor men een fuzzyset opstelt, wordt *linguïstische variabele* genoemd.
Het is een variabele, omdat er verschillende waarden van μ uit afgeleid kunnen worden. Het is een linguïstische variabele, omdat de quantificering ervan niet wiskundig exact is, maar taalkundig gedefinieerd wordt, met alle verwarring en verschillende soorten interpretaties die hiervan het gevolg zijn. Misschien zou dezelfde vraagstelling bij de leden van een Afrikaanse woongemeenschap een heel andere lidmaatschapsfunctie opleveren dan hier in Nederland! Want het is best mogelijk dat het vage begrip "een jonge man" daar heel anders wordt ingevuld.

17.1 Het principe van vage logica (fuzzy logic)

Vereenvoudiging tot lineaire functie
Een grafiekje is een ding, er iets mee kunnen doen heel wat anders! Ook in de vage logica moet gerekend worden, want er moeten immers uitgangsgrootheden worden afgeleid uit de ingangsgrootheden. Rekenen aan ingangsgrootheden kan alleen als het verloop van deze grootheden in exacte wiskundige formules kan worden vast gelegd. Met andere woorden: aan iedere lidmaatschapsfunctie moet een wiskundige vergelijking gekoppeld worden, die de functie zo goed mogelijk beschrijft. De algemene vorm van een dergelijke vergelijking is:
$\mu_A(x)$ = wiskundige uitdrukking
waarbij x een bepaald element uit de vage verzameling is van de linguïstische variabele A. In het voorbeeld zou x bijvoorbeeld gelijk kunnen zijn aan "21 jaar", terwijl A natuurlijk gelijk is aan "een jonge man".
Nu is het vrij ingewikkeld om de Gaussiaanse klokfuctie van het voorbeeld op een simpele manier om te zetten in een wiskundige uitdrukking. Vandaar dat in de meeste gevallen de opgestelde lidmaatschapsfuncties worden vereenvoudigd. De mate van vereenvoudiging wordt de *tolerantie* genoemd. De lidmaatschapsfunctie van de linguïstische variabele "een jonge man" zou bijvoorbeeld vereenvoudigd kunnen worden tot figuur 3/17.1-3. Het zal duidelijk zijn dat deze driehoeksvorm heel wat gemakkelijker in een wiskundige formule omgezet kan worden!
Dit proces noemt men de *lineaire fuzzificering* en de daaruit volgende lidmaatschapsfuncties *lineaire functies.* Nogal logisch, want de lidmaatschapsfunctie bestaat nu nog slechts uit rechte lijntjes. Nu is de tolerantie in het genomen voorbeeld natuurlijk vrij groot. Men kan het ook iets nauwkeuriger doen, door de Gaussiaanse klokfunctie te benaderen door middel van drie of zelfs zeven rechte lijnstukken, die de curve zo goed mogelijk benaderen. Dan ontstaan de LF's die voorgesteld worden in figuur 3/17.1-4.



Figuur 3/17.1-3: De lidmaatschapsfunctie van de linguïstische variabele "een jonge man" wordt vereenvoudigd tot een driehoekvorm.



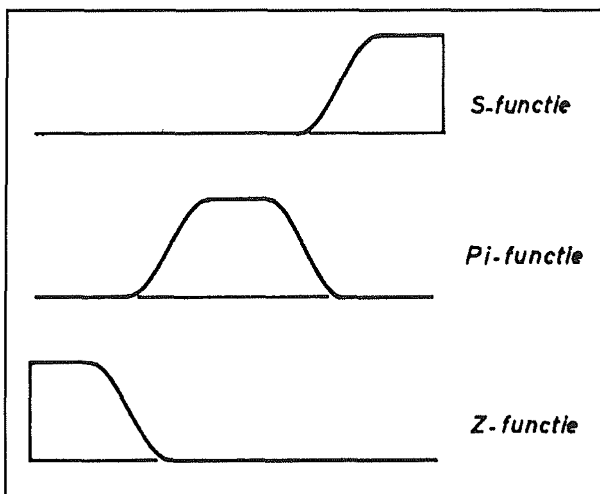
Figuur 3/17.1-4: De tolerantie neemt af als men werkt met gesegmenteerde lineaire lidmaatschapsfuncties.

Dergelijke benaderingen noemt men *gesegmenteerde lineaire lidmaatschapsfuncties.* De oorsprong van deze benaming is duidelijk: de functie bestaat nu immers uit een aantal rechte segmenten.

17.1 Het principe van vage logica (fuzzy logic)

S-, PI- en Z-functies

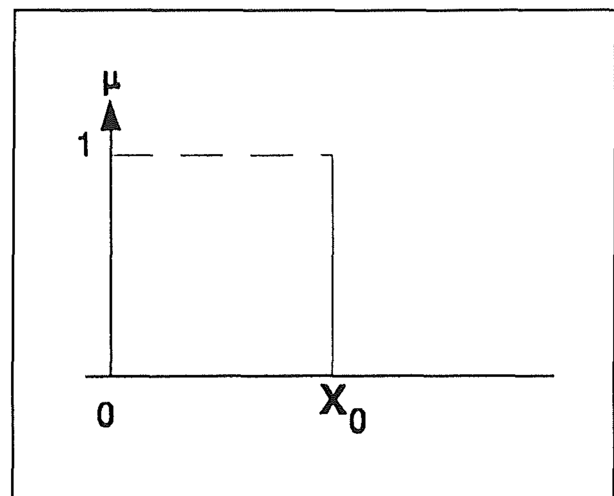
Naast de al dan niet gesegmenteerde lineaire lidmaatschapsfuncties worden in de praktijk ook zogenoemde kwadratische functies toegepast. Er zijn drie standaardvormen, getekend in figuur 3/17.1-5. De S-, PI- en Z-functie kunnen alle drie met dezelfde vrij eenvoudige wiskundige kwadratische vergelijkingen beschreven worden.



Figuur 3/17.1-5: De drie standaardvormen van de kwadratische lidmaatschapsfunctie.

De singleton

Een speciale lineaire lidmaatschapsfunctie is de *singleton.* Een singleton, voorgesteld in figuur 3/17.1-6, heeft slechts één lid van de vage verzameling, waarvoor µ een waarde heeft die niet nul is. Uit de aard der zaak moet deze waarde dan gelijk zijn aan 1. In feite is er aan een singleton niets vaags te ontdekken, en zou een dergelijke verzameling zich heel goed thuis voelen in de harde logica. Het begrip singleton is in de vage logica ingevoerd, omdat men nu eenmaal soms te maken heeft met variabelen die maar één waarde kunnen hebben. Een typisch voorbeeld van een linguïstische variabele, die alleen maar door middel van een singleton functie is uit te drukken is “noem de dag van de week die overeen komt met de datum 5 januari 1996". Hierop is maar één antwoord mogelijk, namelijk "vrijdag". Als figuur 3/17.1-6 de lidmaatschapsfunctie van deze linguïstische variabele zou voorstellen, dan zou X_0 gelijk zijn aan het antwoord “vrijdag”. Uit de aard der zaak is lidmaatschapsgraad μ_{X0} gelijk aan 1.



Figuur 3/17.1-6: De singleton is een lidmaatschapsfuctie, waarvan μ_x slechts twee waarden heeft: 0 of 1.

Het invoeren van termen

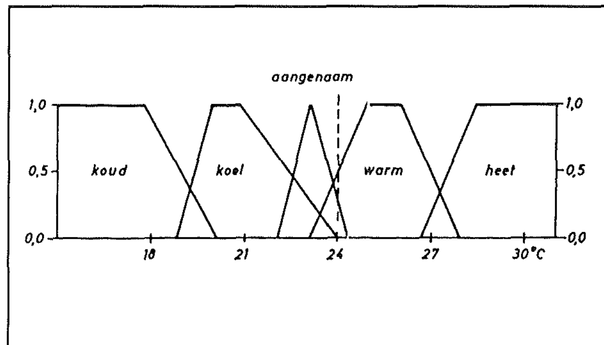
De tot nu toe behandelde lidmaatschapsfuncties zijn in de praktijk niet erg bruikbaar. Spannender word het, als men het begrip *termen* gaat invoeren. Termen zijn bijvoeglijke naamwoorden, die aan de linguïstische variabele gekoppeld worden. Ook dit begrip kan het best aan de hand van een voorbeeld uitgediept worden. Stel de linguïstische variabele “kamertemperatuur”. Dat is een zeer vaag begrip, dat echter beter gedefinieerd kan worden door er enige bijvoeglijke naamwoorden

17.1 Het principe van vage logica (fuzzy logic)

of termen aan te koppelen. Men zou de kamertemperatuur kunnen omschrijven als:

- koud;
- koel;
- aangenaam;
- warm;
- heet.

Als men nu aan duizend personen een lijstje zou overhandigen met daarin ingevuld temperaturen van 10 °C tot 32 °C en zou vragen aan iedere temperatuur een van de bovenstaande vijf termen te koppelen, dan zouden vijf vage verzamelingen ontstaan, die allemaal op de beschreven manier omgezet kunnen worden in een lidmaatschapsfunctie. Men kan deze vijf LF's verzamelen in één grafiek, met als gevolg dat figuur 3/17.1-7 ontstaat.



Figuur 3/17.1-7: De lidmaatschapsfuncties van vijf termen van de linguïstische variabele "kamertemperatuur".

Merk op dat bepaalde temperaturen nu deel uit maken van verschillende verzamelingen. Dat is een zeer fundamenteel verschil tussen de harde logica en de vage logica! De temperatuur "24 °C" maakt bijvoorbeeld deel uit van de verzameling "aangenaam", maar ook van de verzameling "warm". Aan deze temperatuur kan men dus een $\mu_{aangenaam}$ en een μ_{warm} koppelen. In de traditionele harde logica is het absoluut ondenkbaar dat een element deel uitmaakt van twee verzamelingen.

De kracht van de vage logica

Het feit dat een element deel kan uitmaken van meer dan één vage verzameling is de kracht van de vage logica. Door linguïstische variabelen om te zetten in lidmaatschapsfuncties van een aantal termen, ontstaat een zeer verfijnd beeld van de variabele, een beeld dat als het ware op een menselijke manier met de variabele omgaat. Hierdoor kunnen regelsystemen als het ware natuurlijk afgestemd worden op het menselijk gedrag.

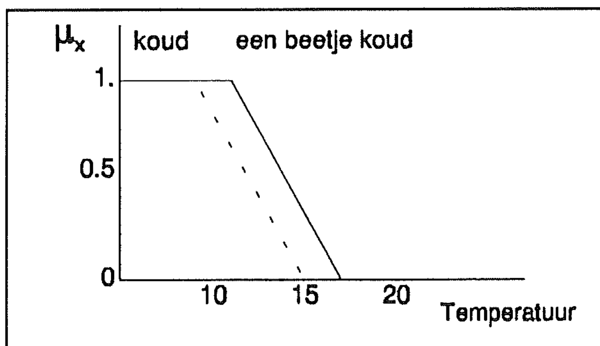
Het wezenlijke verschil tussen harde logica en vage logica kan het best aan de hand van een voorbeeld worden toegelicht. Stel dat men een kleurenfoto wil omzetten in een monochroom beeld. Volgens de regels van de harde logica zou een beeld ontstaan, dat alleen is opgebouwd uit beeldcellen die ofwel wit ofwel zwart zijn. Volgens de regels van de vage logica zou een beeld ontstaan waar, naast echt wit en echt zwart, ook nog honderden grijstinten aanwezig zijn. Het zal duidelijk zijn dat wij, menselijke wezens, het grijstinten beeld veel beter zouden kunnen interpreteren dan het zwart/wit-beeld. Voor een computer ligt dat anders. Als de monochrome beelden gebruikt zouden moeten worden om via de computer aan beeldherkenning te doen, dan zou de computer veel gemakkelijker de keiharde zwart/wit-beelden kunnen evalueren dan de grijstint afbeeldingen.

Het invoeren van hedges

De theorie van de vage logica heeft nog een tweede manier bedacht om problemen nauwkeuriger te fuzzificeren. Dat zijn de *hedges*. In taalkundige termen uit-

17.1 Het principe van vage logica (fuzzy logic)

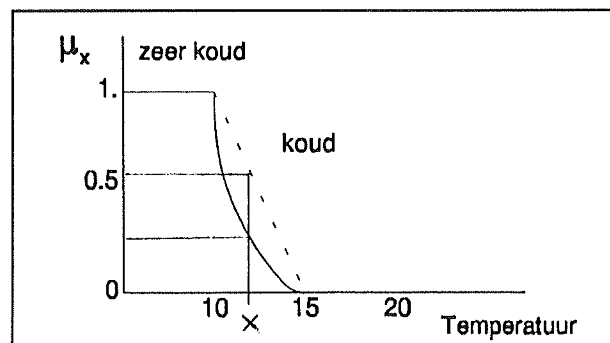
gedrukt zou men hedges kunnen opvatten als bijwoorden, die aan de termen van de linguïstische variabele worden gekoppeld. De term "koel" van de linguïstische variabele "kamertemperatuur" zou bijvoorbeeld nader gespecificeerd kunnen worden met de hedges "een beetje" en "zeer".
Er ontstaan dan twee tamelijk nauwkeurige linguïstische begrippen, namelijk "een beetje koele kamertemperatuur" en "zeer koele kamertemperatuur".
Dank zij deze hedges kan de ontwerper van een vaag regelsysteem ingrijpen in de lidmaatschapsfuncties. Het zal immers duidelijk zijn dat het invoeren van hedges invloed heeft op de vorm van de lidmaatschapsfuncties. Hedges worden vaak ingevoerd in de testfase van een vaag regelsysteem. Als dan blijkt dat de opgestelde lidmaatschapsfuncties van de in- en uitgangsgrootheden toch niet het gewenste regeleffect veroorzaken, dan kan men het systeem verfijnen door de lidmaatschapsfuncties iets aan te passen door het invoeren van een of meerdere hedges.
Hiervoor zijn twee benaderingen mogelijk:
- shifted hedges;
- powered hedges.



Figuur 3/17.1-8: De werking van een shifted hedge op de lidmaatschapsfunctie van een term: een verschuiving over de as.

Shifted hedges
Een shifted hedge doet niets anders dan de grenzen van de lidmaatschapsfunctie van de term naar links of naar rechts verschuiven. In figuur 3/17.1-8 is aangegeven hoe de hedge "een beetje" de LF van de term "koud" uit figuur 3/17.1-7 naar rechts verschuift.

Powered hedge
Bij de powered hedge worden de grenzen van de LF niet verschoven, maar wordt de vorm ervan aangepast. In figuur 3/17.1-9 is getekend hoe door het invoeren van de powered hegde "zeer" in de term "koud" de lidmaatschapsfunctie zo wordt aangepast dat lagere temperaturen in de verzameling "koud" een minder dan lineaire toename van hun μ krijgen.



Figuur 3/17.1-9: De invloed van een powered hedge.

Een bepaalde temperatuur X heeft zonder de hedge een μ_x van ongeveer 0,5. Door het invoeren van de powered hedge daalt zijn μ tot ongeveer 0,25.

Het redeneermechanisme

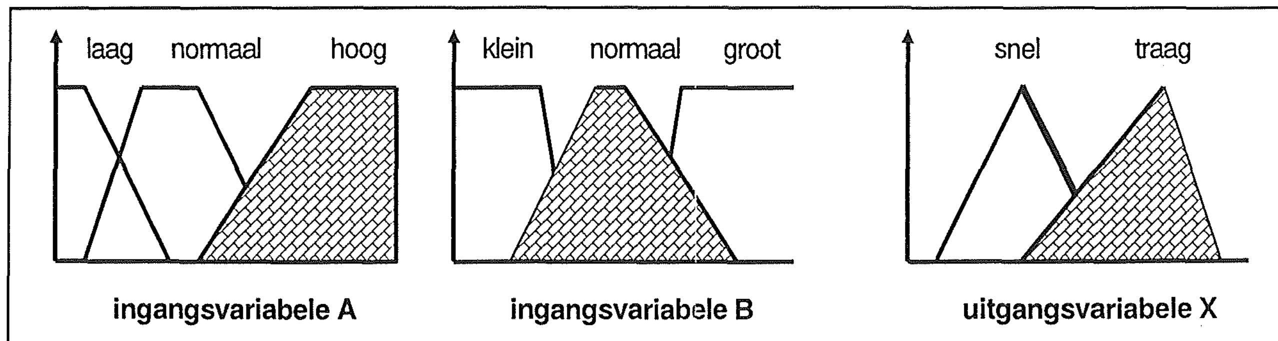
Inleiding
Doel van het redeneermechanisme is een aantal regels opstellen, die de gewenste

17.1 Het principe van vage logica (fuzzy logic)

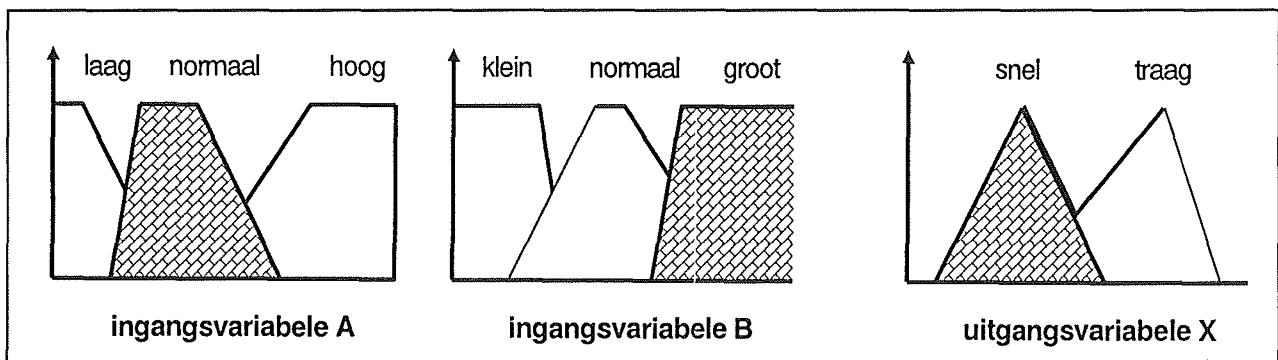
relatie tussen de ingangs- en de uitgangsgrootheden vast leggen. In principe wijkt deze stap niet af van het opstellen van de beslissingsregels in de harde logica. Het enig verschil is dat de beslissingsregels in de vage logica minder hard zijn en meer aangepast aan de menselijke subjectieve maatstaven.
De meest algemene vorm van een vage beslissingsregel is:
ALS *premisse* DAN *conclusie*
De premisse bestaat uit een of meer uitspraken, *antecendenten* genoemd, die betrekking hebben op de ingangsgrootheden. Een voorbeeld van een premisse is:
"kamertemperatuur IS warm EN buitentemperatuur IS koel".
Twee linguïstische ingangsvariabelen "kamertemperatuur" en "buitentemperatuur" worden hierin nader omschreven door hun termen "warm" en "koel" en door middel van een fuzzy-operator EN aan elkaar gekoppeld. De premisse geeft dus als het ware aan welke lidmaatschapsfuncties van de ingangsgrootheden in de regel actief zijn. Bij het uitwerken van deze regel zal de fuzzy-processor de actuele waarden van de twee ingangsgrootheden meten, deze op de assenstelsels van de LF's zetten en de μ's berekenen van de actieve lidmaatschapsfuncties. Deze twee waarden van μ worden dan gebruikt om de uitgangsvariabele te berekenen.
De *conclusie* heeft op dezelfde manier betrekking op de uitgangsgrootheden. In het voorbeeld:
motorsnelheid IS hoog
wordt de LF van term "hoog" van de linguïstische uitgangsvariabele "motorsnelheid" geactiveerd. De berekende μ's van de actieve ingangs-LF's worden aan de geselecteerde uitgangs-LF aangeboden en er wordt op een bepaalde manier een specifieke waarde voor de uitgangsvariabele berekend. Op dit rekenmechanisme wordt later nader ingegaan.

Voorbeeld
Aan de hand van een voorbeeldje wordt de praktische betekenis van premisse en conclusie toegelicht.
Stel dat een bepaald systeem een uitgangsvariabele X heeft, die op een bepaalde manier moet reageren op de waarde van twee ingangsvariabelen A en B. Volgens de regels van de fuzzy kunst worden deze drie linguïstische grootheden nader gespecificeerd door termen. A kan ingedeeld worden in de drie termen "laag", "normaal" en "hoog". Variabele B krijgt ook drie termen, namelijk "klein", "normaal" en "groot". De uitgangsvariabele X kan met de beste wil van de wereld niet beter dan met de twee termen "snel" en "traag" beschreven worden. Vervolgens stelt men voor de acht termen lidmaatschapsfuncties op en tekent deze in drie grafieken. Een en ander is toegelicht in figuur 3/17.1-10.
Vervolgens stelt men een vage beslissingsregel op:
ALS A IS hoog EN B IS normaal DAN X IS TRAAG
Het fuzzy-systeem weet nu dat bij de uitwerking van deze regel het rekening moet houden met de LF van "A-hoog" en de LF van "B-normaal". De overige LF's van de ingangsvariabelen zijn niet aan de orde. Het resultaat zal uitgelezen moeten worden in de LF van "X-traag". Hoe dat in- en uitlezen gaat wordt later besproken.
Men kan nu voor dit systeem een tweede vage beslissingsregel opstellen, bijvoorbeeld:
ALS A IS normaal EN B IS groot, DAN X is snel
In figuur 3/17.1-11 is getekend welke lidmaatschapsfuncties nu actief worden.

17.1 Het principe van vage logica (fuzzy logic)



Figuur 3/17.1-10: Grafische toelichting van de reële betekenis van de premisse en de conclusie.



Figuur 3/17.1-11: De actieve LF's voor de vage regel "Als A is normaal EN B is groot, DAN X is snel".

Verfijningen

Uiteraard stelt de theorie een aantal verfijningen in, waardoor een beslissingsregel veel nauwkeuriger het verband tussen in- en uitgangsgrootheden kan vastleggen. De voornaamste verfijningen zijn:
- fuzzy-operatoren;
- evaluerende regels;
- voorspellende regels.

Fuzzy-operatoren

Net zoals de harde Booliaanse algebra kent ook het rekenstelsel van de fuzzy logic een aantal operatoren, te weten:
- EN (AND, UND);
- OF (OR, ODER);
- NIET (NOT, NICHT).

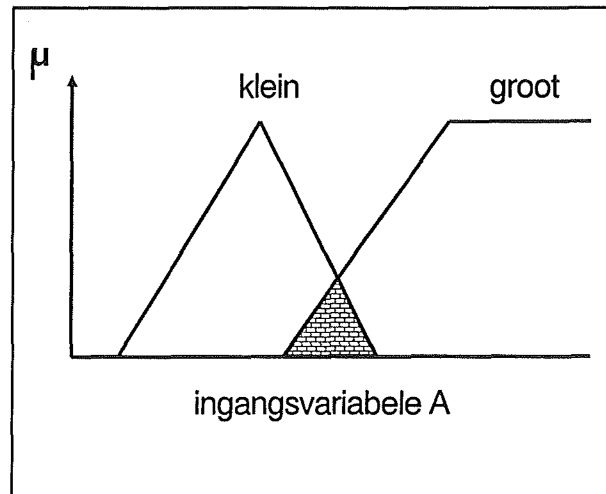
Men kan deze operatoren opnemen in een premisse om de onderlinge relatie tussen de ingangsgrootheden beter te omschrijven.

De EN-operator

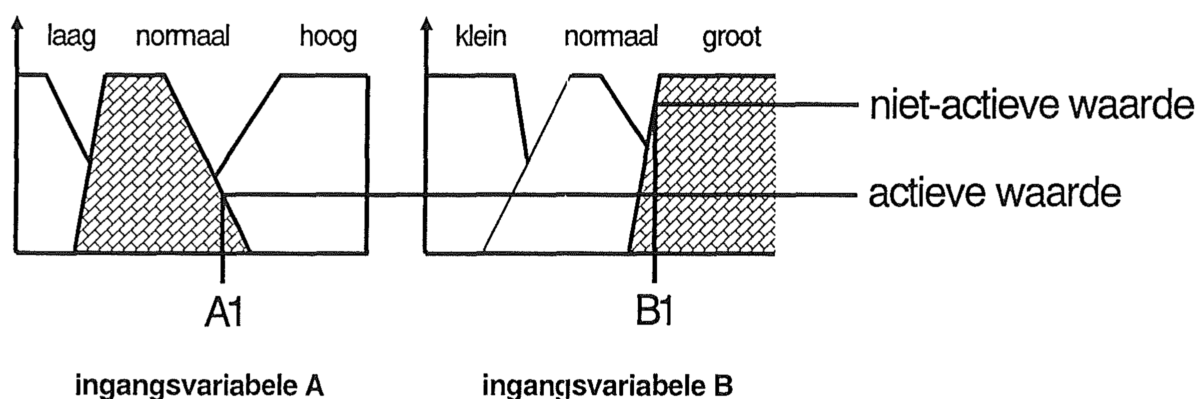
De EN-operator laat toe twee lidmaatschapsfuncties van een variabele actief te maken in één beslissingsregel. Zoals uit figuur 3/17.1-12 blijkt, zal bij een EN-koppeling alleen het gemeenschappelijk oppervlak van beide LF's actief worden. De figuur geeft het resultaat van:
ALS A IS klein EN A is groot DAN . . .
Natuurlijk wordt de EN-operator ook gebruikt om twee ingangsvariabelen met elkaar te koppelen, zoals in:
Als A is groot EN B IS klein DAN . . .
In dit geval zal de kleinste μ-waarde van beide LF's in de berekening betrokken worden. Dit wordt toegelicht aan de hand

17.1 Het principe van vage logica (fuzzy logic)

van figuur 3/17.1-13. De vage beslissingsregel van dit voorbeeld luidt:
ALS A IS normaal EN B IS groot DAN . . .
Op het moment dat de fuzzy-processor deze regel verwerkt, meet hij de actuele waarden van de ingangsvariabelen. Deze zijn A1 en B1. Deze worden op de assen van de functies gezet. Hieruit kan men twee actieve μ-waarden berekenen, een voor A1 (μ_{A1}) en een voor B1 (μ_{B1}). Omdat A en B in de regel door een EN-operator gekoppeld zijn, zal nu alleen de laagste μ-waarde overgedragen worden naar de uitgangsfunctie voor het berekenen van de actuele waarde van de uitgang. In dit voorbeeld is dit dus duidelijk μ_{A1}.



Figuur 3/17.1-12: Het resultaat van een EN-operator op twee lidmaatschapsfuncties van een variabele.

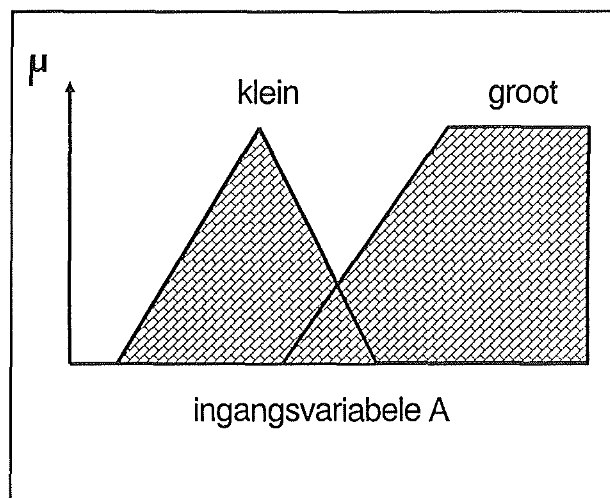


Figuur 3/17.1-13: De invloed van een EN-operator op het evalueren van de waarden van de premisse.

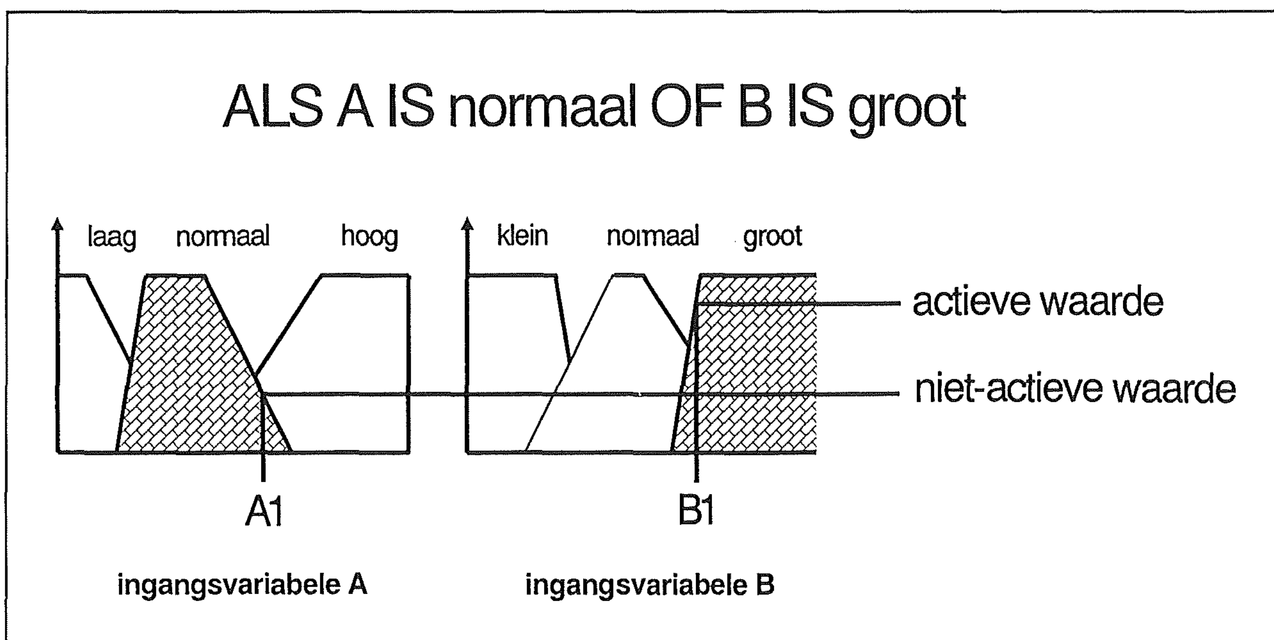
17.1 Het principe van vage logica (fuzzy logic)

De OF-operator
Ook deze operator laat toe twee LF's van een ingangsvariabele te activeren. In dit geval wordt, zoals getekend in figuur 3/17.1-14, het gezamenlijk oppervlak van beide LF's geactiveerd. De premisse van dit voorbeeld luidt:
ALS A IS klein OF A IS groot DAN . . .
Natuurlijk kan men de OF-operator ook gebruiken om twee variabelen te koppelen. In figuur 3/17.1-15 is als voorbeeld de premisse:
ALS A IS normaal OF B IS groot DAN . . .
Ook nu meet de fuzzy-processor de actuele waarden van de ingangen en berekent uit de actieve LF's hun μ-waarden. Maar vanwege de OF-koppeling wordt nu de grootste μ-waarde doorgekoppeld naar de uitgangsfunctie. In dit geval is dit dus μ_{B1}.

De NIET-operator
De NIET-operator is de ook in de harde logica bekende inverteer-functie. Het resultaat van een NIET-operatie is dat alle waarden van μ opnieuw worden berekend volgens de formule:
$\overline{\mu}_A(X) = 1 - \mu_A(X)$
Met andere woorden: de geïnverteerde waarden $\overline{\mu}$ van μ kan men berekenen door de waarden van μ van 1 af te trekken. Wat dit grafisch betekent is voorgesteld in figuur 3/17.1-16.

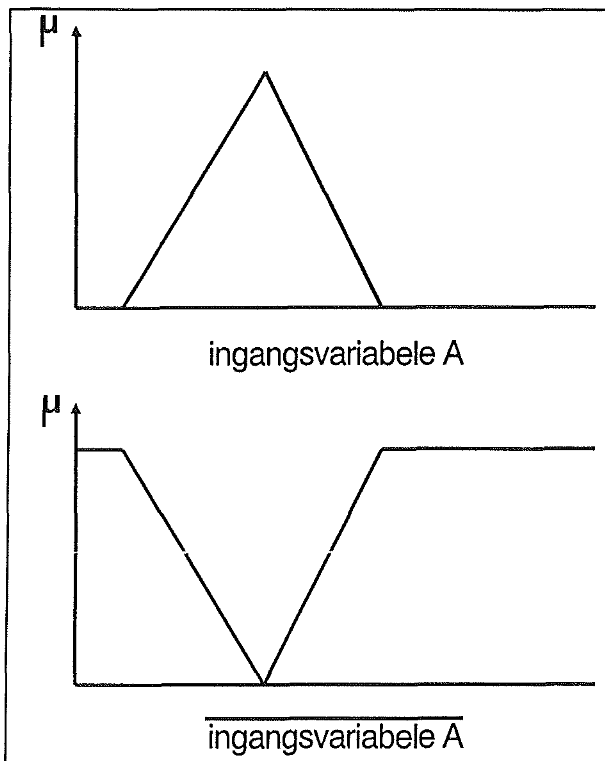


Figuur 3/17.1-14: Het koppelen van twee LF's door middel van een OF-operator.



Figuur 3/17.1-15: Het koppelen van twee ingangsvariabelen door middel van een OF-operator.

17.1 Het principe van vage logica (fuzzy logic)



Figuur 3/17.1-16: Het resultaat van een NIET-bewerking op een lidmaatschapsfunctie.

Evaluerende regels
Evaluerende regels worden het vaakst toegepast. Alle tot nu toe gebruikte voorbeelden zijn evaluerende regels. De meest algemene uitdrukking van een evaluerende regel is:
ALS A IS a EN/OF B IS b DAN X is x
waarin staan A en B voor ingangsvariabelen, a en b voor hun actieve termen, X voor de uitgangsvariabele en x voor zijn actieve term.
De meeste opdrachten uit de vage logica kunnen uitstekend met evaluerende regels opgelost worden.

Voorspellende regels
Voorspellende regels zijn niet zo gemakkelijk op te stellen, maar zij bieden wel zeer krachtige functies.

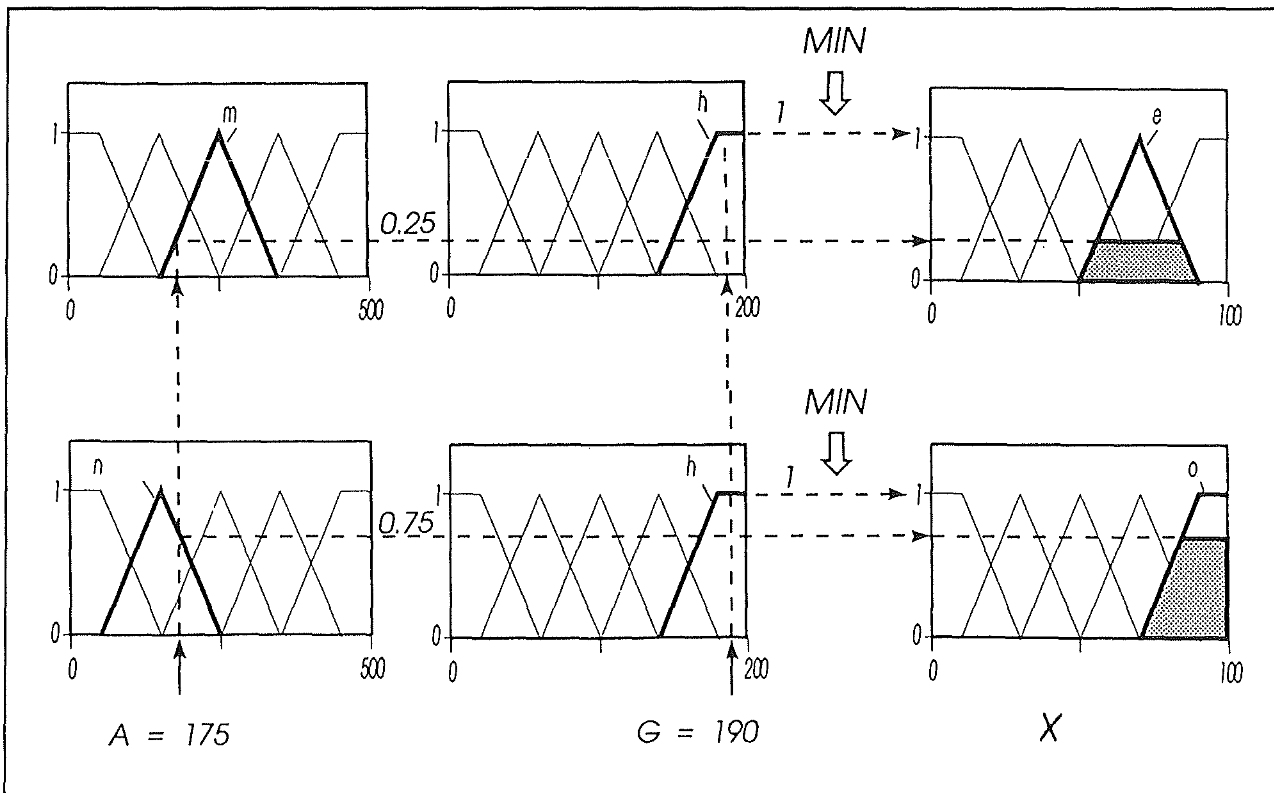
Een voorbeeld van een voorspellende regel, zuiver onder linguïstische vorm is:
"**ALS** krachtiger remmen bij een gegeven snelheid tot gevolg heeft dat de motor sneller stopt binnen de toegestane afstand, ga **DAN** krachtige remmen"
Een specifieke eigenschap van voorspellende regels is dat de uitgangsvariabele altijd wordt opgenomen binnen de premisse van de regel. Er kunnen een aantal voorspellende regels onder elkaar in het programma worden opgenomen. De fuzzy-processor zal deze dan een voor een evalueren en de voorspellende regel die het beste stuursignaal oplevert vervolgens selecteren en uitvoeren.
In termen van fuzzy notatie kan een voorspellende regel als volgt geschreven worden:
ALS
  [X IS x ⇒ (A IS a EN/OF B IS b)]
DAN
  X IS x

De inferentie

Inleiding
De *inferentie* is de techniek van het redeneer mechanisme, waarmee de fuzzy-processor uit de opgestelde regels een bepaalde uitgangsfunctie afleidt. Deze functie hangt bovendien af van de momentele waarden van de μ's van de ingangen, maar is nog steeds een vage verzameling die men een fuzzy-functie noemt.
Een voorbeeld zal het inferentie-proces verduidelijken.

Voorbeeld
- Gegeven:
  een regelsysteem met twee ingangen A en G en een uitgang X

17.1 Het principe van vage logica (fuzzy logic)



Figuur 3/17.1-17: Het principe van inferentie toegelicht aan de hand van een voorbeeld.

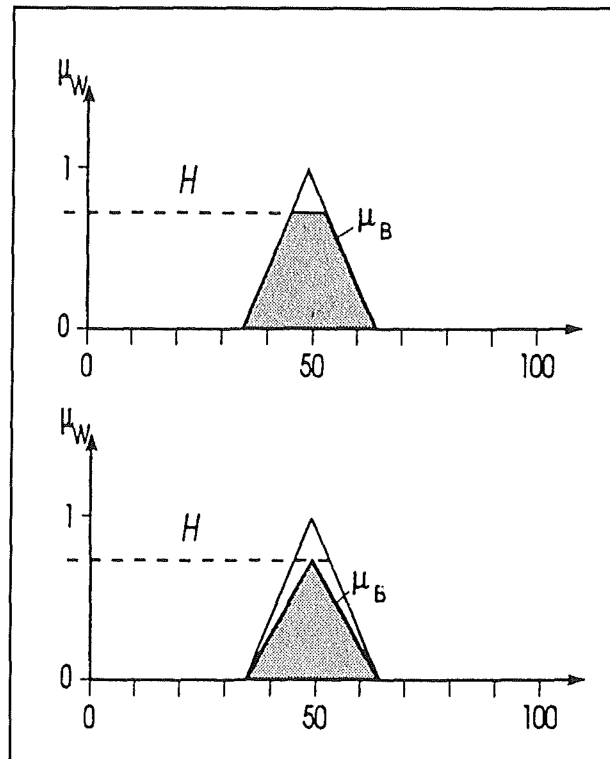
- Fuzzicatie:
  - ingang A kan beschreven worden in vijf termen, waaronder m en n;
  - ingang G kan beschreven worden in vijf termen, waaronder h;
  - uitgang X kan beschreven worden in vijf termen, waaronder e en o.
- Redeneermechanisme:
  Het te sturen proces kan volledig beschreven worden met twee vage regels:
  - Regel 1:
    ALS A IS m En G IS h DAN X is e
  - Regel 2:
    ALS A IS n EN G IS h DAN X IS o
- Actuele ingangssituatie:
  Op het moment dat de fuzzy-processor de regels uitwerkt is de actuele waarde van de ingangsgrootheden:
  - A = 175
  - G = 190

Het volledige proces van inferentie is samengevat in figuur 3/17.1-17.
De bovenste grafiek geeft de inferentie van de eerste regel. Natuurlijk zijn de LF's m van A en h van G actief, want die staan vermeld in de eerste regel. Uit de actuele waarden 175 en 190 kunnen de twee corresponderende µ-waarden berekend worden, waaruit blijkt dat $\mu_A = 0{,}25$ en $\mu_G = 1$. Omdat beide variabelen door middel van een EN-operator gekoppeld zijn, telt alleen de laagste µ-waarde, in dit geval 0,25. Deze wordt horizontaal doorgetrokken naar de grafiek van de uitgang X. Hier is de LF van e actief. Er ontstaat nu een nieuwe vage verzameling, die begrensd wordt door de µ-waarde 0,25. Deze fuzzy-functie is gearceerd weergegeven.
De onderste grafiek geeft de inferentie van de tweede regel. Nu is de LF A_n actief, alsmede de LF van G_h. De waarden van de

ingangen zijn natuurlijk nog niet gewijzigd, met als gevolg dat er nu twee μ-waarden berekend kunnen worden van 0,75 en 1. Ook nu wordt er gekoppeld met een EN-operator, zodat de laagste μ (0,75) weer wordt doorgetrokken naar de uitgangsgrafiek. Hier is de LF X_0 actief. Ook nu berekent het inferentie-proces een nieuwe vage verzameling, die nu begrensd wordt door de μ-waarde 0,75.

Soorten inferentie
Er zijn verschillende algoritmen ontwikkeld voor het opstellen van fuzzy-functies door middel van inferentie. De meest toegepaste zijn:
- MIN-MAX inferentie;
- MAX-PROD inferentie.

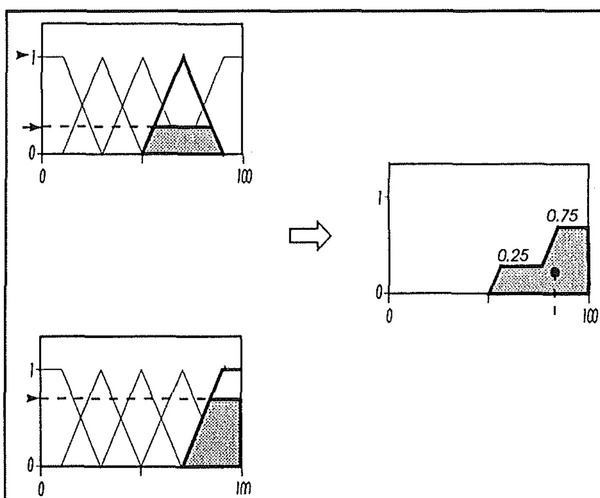
De verschillen worden toegelicht aan de hand van figuur 3/17.1-18.
De bovenste tekening geeft het resultaat van een MIN-MAX inferentie. De actieve LF van de uitgang wordt begrensd door de actieve waarde van de μ van de ingangen. Hieruit volgt dat het inferentie voorbeeld van figuur 3/17.1-17 volgens het MIN-MAX algoritme werkt.
De onderste tekening toont wat er gebeurt als wordt gewerkt volgens het MAX-PROD schema. Nu wordt de actieve LF van de uitgang niet begrensd, maar verkleint in hoogte, tot de top samenvalt met de actieve waarde van μ.
Er is geen sprake van een "goede" inferentie-strategie. Voor sommige toepassingen kan men het best gebruik maken van de MIN-MAX algoritmen, andere toepassingen geven de beste resultaten als men volgens MAX-PROD werkt.
Bovendien zijn er tientallen andere inferentie-strategieën theoretisch uitgewerkt, die weliswaar nog niet vaak worden toegepast in reële toepassingen van fuzzy techniek, maar wat niet is kan nog komen!



Figuur 3/17.1-18: Twee vormen van inferentie-strategie: boven MIN-MAX, onder MAX-PROD.

De compositie
Door de inferentie wordt iedere regel van het redeneer mechanisme omgezet in een fuzzy-functie, die het door de uitgang te volgen gedrag definieert bij bepaalde waarden van de ingangsvariabelen. Natuurlijk heeft men daar in de praktijk vrij weinig aan, want wat moet ontstaan is een eenduidige uitgangsfunctie, waaruit een welbepaalde waarde voor de uitgang kan worden afgeleid. Op de een of andere manier moeten alle afzonderlijke fuzzy-functies van de uitgang worden gecombineerd tot één algemeen geldende functie, die rekening houdt met alle stellingen die in alle regels vast gelegd zijn. Dit proces noemt men de *compositie.*
In de meeste gevallen gaat dit door de afzonderlijke fuzzy-functies met elkaar te verknopen met een OR-operator. Dat is

vrij logisch, want men kan er van uitgaan dat het systeem aan alle regels van het redeneer mechanisme moet voldoen, hetgeen te vertalen is naar een OR-functie. In het voorbeeld van figuur 3/17.1-17 levert de compositie dus een eenduidige uitgangs-functie op, die ontstaat door de twee deel-functies te OR-ren. Hetgeen, zoals reeds beschreven, er op neer komt dat de oppervlakken van de deelfuncties tot één oppervlak worden verenigd. Een en ander wordt grafisch toegelicht in figuur 3/17.1-19.



Figuur 3/17.1-19: Door een OR-operator toe te passen op de twee deel-functies ontstaat de uiteindelijke uitgangs-functie van het fuzzy-proces.

De defuzzificatie

Inleiding

De uitgangs-functie die door de compositie ontstaat bepaalt volledig de reactie van de uitgang op de momentele ingangswaarden. Natuurlijk moet uit deze functie een expliciete waarde van de uitgangsvariabele worden afgeleid. Dit proces noemt men de *defuzzificatie.*

Ook hiervoor zijn verschillende strategieën ontwikkeld, waarvan de voornaamste zijn:

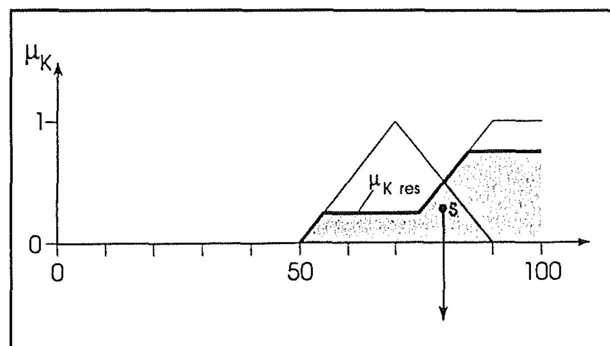
- de center of gravity strategie COG;
- de maximum strategie MAX;
- de mean of maximum strategie MOM.

De center of gravity strategie wordt het vaakst toegepast.

De center of gravity strategie

Bij deze strategie bepaalt de fuzzy-processor het zwaartepunt S van de uitgangsfunctie. Het zwaartepunt of *centroide* is een wiskundig begrip, dat soms met eenvoudige en soms met ingewikkelde wiskunde op ieder plat vlak kan worden toegepast. In figuur 3/17.1-20 is getekend waar het zwaartepunt van de uitgangsfunctie van figuur 3/17.1-20 ligt. Door uit dit zwaartepunt S een vertikaal hulplijntje te tekenen naar de horizontale as kan de numerieke waarde van de uitgangsvariabele afgelezen worden.

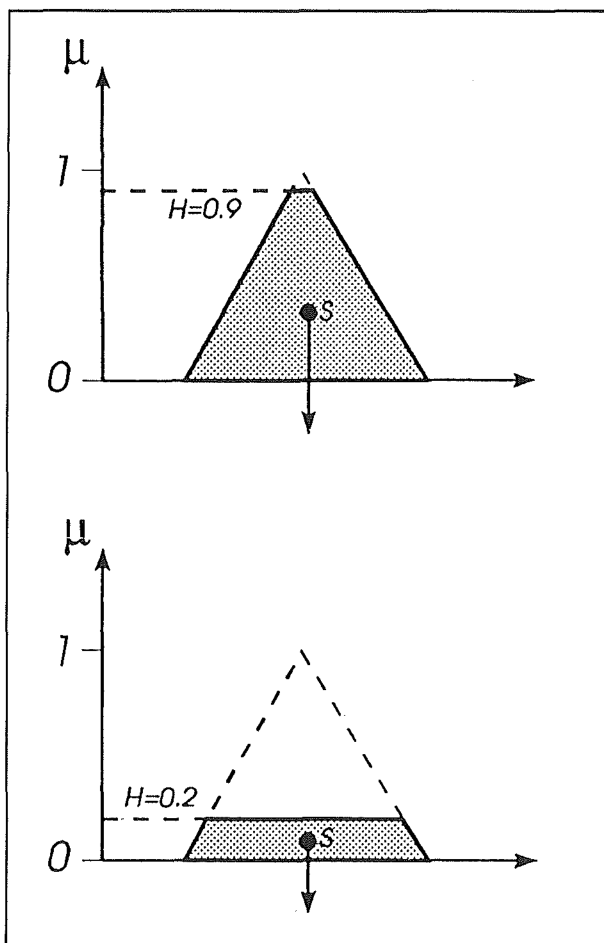
In het voorbeeld betekent dit dat het fuzzy systeem de uitgangsvariabele X een waarde van 80 geeft, als de ingangsvariabele A gelijk is aan 175 en de ingangsvariabele G gelijk is aan 190.



Figuur 3/17.1-20: Het bepalen van de numerieke waarde van de uitgangsvariabele door middel van de COG-strategie.

17.1 Het principe van vage logica (fuzzy logic)

De COG-strategie heeft een groot nadeel, dat toegelicht wordt aan de hand van figuur 3/17.1-21.
In die tekening zijn twee uitgangs-functies getekend, die duidelijk een andere vorm hebben. Als men van beide functies het zwaartepunt berekent en daarvan de horizontale waarde, dan blijkt dat deze waarden gelijk zijn! In beide gevallen zou de uitgangsvariabele dus dezelfde numerieke waarde krijgen, terwijl het toch duidelijk is dat de ingangsvariabelen niet gelijk zijn!

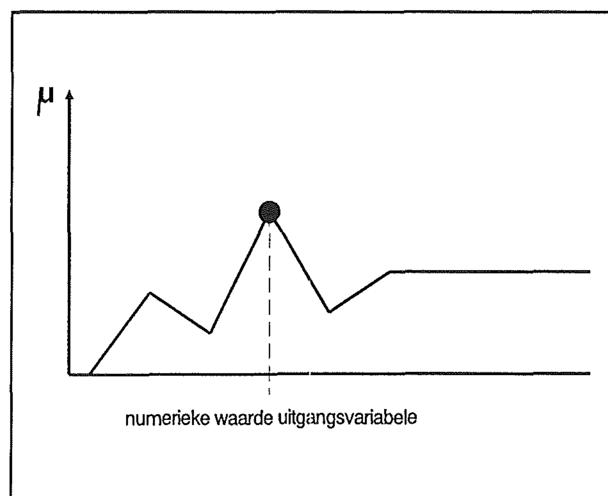


Figuur 3/17.1-21: Een nadeel van de COG-strategie is dat twee verschillende uitgangs-functies dezelfde waarde voor de uitgang kunnen opleveren.

Ondanks dit bezwaar schijnt de COG-strategie in de praktijk zeer goede resultaten op te leveren.
Een tweede nadeel van de COG-strategie is dat het bepalen van het zwaartepunt bij ingewikkelde, bijvoorbeeld niet-lineaire uitgangs-functies een hele klus is, waar zelfs een snelle processor de nodige tijd over doet.

De maximum strategie
Bij de MAX-strategie wordt het punt van de uitgangs-functie bepaald dat de hoogste waarde op de µ-as heeft. Vanuit dit punt wordt weer een vertikale lijn naar de horizontale as getrokken, waar de numerieke waarde van de uitgangsvariabele afgelezen kan worden. In figuur 3/17.1-22 is dit toegelicht aan de hand van een voorbeeldje. Voordeel van de MAX-strategie is dat het bepalen van het maximum van een curve wiskundig heel snel gaat en dat de processor dus weinig rekentijd nodig heeft voor de defuzzificatie.

De mean of maximum strategie
De MOM-defuzzificatie werkt volgens een heel eigen principe.



Figuur 3/17.1-22: Defuzzificatie volgens de MAX-strategie.

17.1 Het principe van vage logica (fuzzy logic)

Er wordt namelijk géén compositie toegepast. Iedere regel uit het redeneer mechanisme wekt, zoals bekend, een specifieke functie op voor de uitgang. Bij de MOM-strategie wordt van iedere uitgangsfunctie het maximum bepaald en de bijbehorende μ_{max} berekend.
Nadien wordt het gemiddelde van al deze waarde berekend en dit getal wordt gebruikt om de reële waarde van de uitgangsvariabele te bepalen. Het voordeel van deze strategie is dat de fuzzy-processor veel minder hoeft te rekenen en daardoor veel sneller kan werken.

Conclusie

Samenvatting
Hiermee is het gehele fuzzy-proces stap na stap besproken. Een samenvatting lijkt op zijn plaats.
Het oplossen van een regelprobleem door middel van vage logica gaat als volgt:
- Stap 1:
  Bepaal wat de in- en de uitgangsvariabelen zijn.
- Stap 2:
  Bepaal de minimum en maximum grenzen van deze variabelen.
- Stap 3:
  Onderzoek hoeveel termen er nodig zijn om het gedrag van de variabelen te beschrijven.
- Stap 4:
  Stel de lidmaatschapsfuncties voor alle termen op.
- Stap 5:
  Onderzoek hoe de uitgangsvariabele zich moet gedragen in functie van de ingangsvariabelen.
- Stap 6:
  Stel aan de hand hiervan de regels van het redeneer mechanisme op.
- Stap 7:
  Bepaal welke inferentie het best kan worden toegepast.
- Stap 8:
  Bepaal welke defuzzificatie strategie het best kan worden toegepast.

Fuzzy logic in de praktijk
Hoe worden problemen in de praktijk opgelost met vage logica?
Daarvoor bestaan twee benaderingen, namelijk:
- softwarematig;
- hardwarematig.

Het zal wel zonder nadere toelichting duidelijk zijn dat er onder de vage logica een keiharde kern zit, samengesteld uit wiskundige formules en dito berekeningen. Dergelijke formules en berekeningen kunnen natuurlijk door een "normale" processor uitstekend opgesteld en uitgevoerd worden. Vandaar dat het heel goed mogelijk is fuzzy logic op een computer te bedrijven. Het enige dat hiervoor noodzakelijk is, is een geschikt pakket waarmee men in- en uitgangsvariabelen kan vast leggen, de diverse fuzzy-set's kan invoeren en de beslissingsregels kan definiëren. Nadien kan men de computer alle noodzakelijke berekeningen laten uitvoeren om uit een combinatie van ingangsgrootheden een bepaalde uitgangsgrootheid af te leiden.

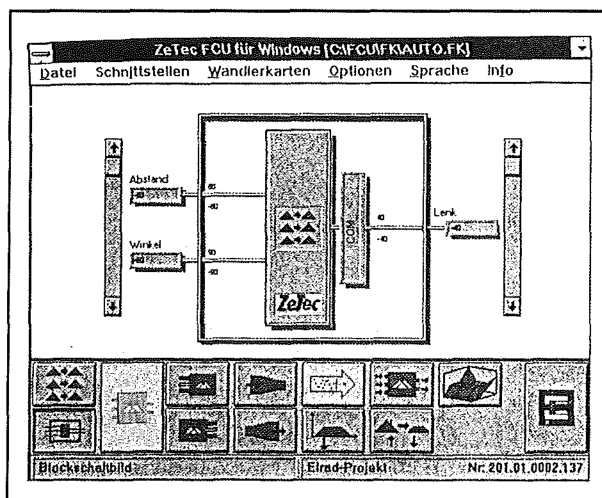
Fuzzy-processoren
Bij de hardwarematige benadering maakt men gebruik van speciale fuzzy-processoren. Daarin zit natuurlijk geen vage elektronica, want zoiets bestaat niet, maar een normaal geheugen waarin men de beslissingsregels kan opslaan en rekentabellen die lidmaatschapsfuncties voorstellen.

17.1 Het principe van vage logica (fuzzy logic)

Dergelijke processoren werken op kloksnelheden tot 20 MHz en zijn in staat in enkele micro- tot enkele milliseconden een volledig fuzzy-proces uit te rekenen. In hoofdstuk 3/17.3 zal iets dieper worden ingegaan op de werking van dergelijke processoren.

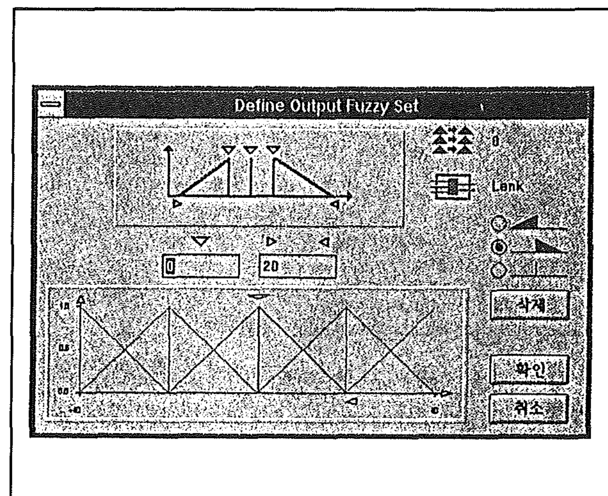
Software emulator

Omdat het (voorlopig) niet de bedoeling is een afzonderlijk hoofdstuk te besteden aan software emulatoren, wordt dit hoofdstuk afgesloten met een korte schets van de manier waarop deze pakketten werken. Als voorbeeld wordt "ZeTec FCU for Windows" behandeld". FCU staat hierbij voor "Fuzzy Construction Unit".
Allereerst moet het regelsysteem in beeld worden gebracht. Dat kan via het in figuur 3/17.1-23 getekende structuurscherm. Hier kan men de in- en uitgangsvariabelen definiëren, hun linguïstische namen invoeren en hun minimum en maximum waarden vast leggen.

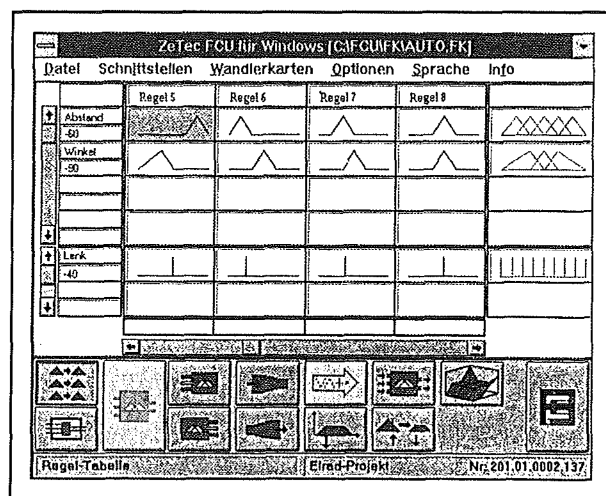


Figuur 3/17.1-23: Het structuurscherm van ZeTec FCU for Windows.

Vervolgens moet men natuurlijk de termen van de variabelen bepalen en hun lidmaatschapsfuncties, hetgeen in ieder geval mensenwerk is. Maar deze lidmaatschapsfuncties kunnen nadien wél eenvoudig via het scherm van figuur 3/17.1-24 in de computer ingevoerd worden.



Figuur 3/17.1-24: Het invoeren van de lidmaatschapsfuncties.



Figuur 3/17.1-25: Het invoeren van de beslissingsregels.

Het opstellen van de beslissingsregel is natuurlijk ook weer mensenwerk, maar via het scherm van figuur 3/17.1-25 kunnen deze op een heel gebruikersvriendelijke manier in de computer ingevoerd worden.

17.1 Het principe van vage logica (fuzzy logic)

Nadien moet men opgeven welke defuzzificerings-strategie men wil toepassen en natuurlijk de ingangsgegevens invoeren. Dat kan met de hand, maar de meeste systemen laten ook toe gegevens in te lezen via de seriële poorten van de computer.

17.1 Het principe van vage logica (fuzzy logic)

3/17.2

Voorbeelden van regelsystemen met vage logica

Inleiding

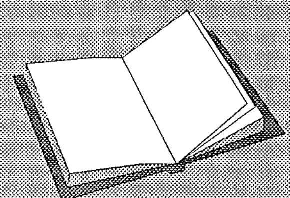
Waar en wanneer?
Vage logica wordt voornamelijk toegepast bij ingewikkelde regelsystemen, waar de harde logica op de grenzen van haar mogelijkheden stuit.
Of, bij systemen waar in de praktijk is gebleken dat vage logica verfijndere regelingen tot stand brengt dan harde logica.
Om een indruk te geven van de veelzijdige mogelijkheden van de vage logica worden in dit hoofdstuk enige voorbeelden besproken van al dan niet experimentele regelsystemen, waar gebruik wordt gemaakt van vage logica.
Bewust is gekozen voor ver uiteenlopende toepassingsgebieden, zodat duidelijk wordt hoe universeel toepasbaar vage logica is.
Bovendien zal uit de besproken voorbeelden heel duidelijk worden, dat het ontwerpen van vage logica in feite helemaal niet zo moeilijk is als het lijkt.

Voorbeelden
De volgende voorbeelden worden behandeld:
- besturing van een robot-arm;
- besturing van het roer van een schip;
- kwaliteitsbewaking van de basisgrondstof voor de fabricage van rubber produkten.

De robot-arm

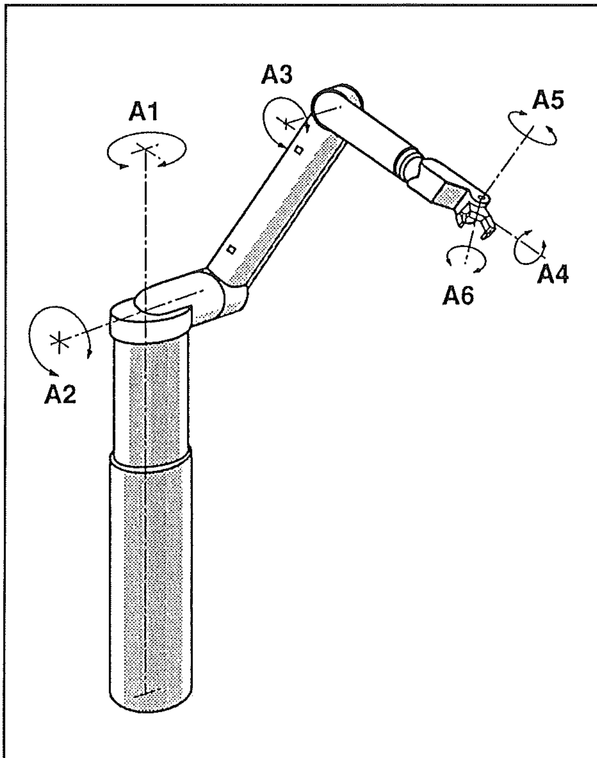
Inleiding
Door TNO werd een robot-arm ontwikkeld met als toepasselijke naam “Manus”. De bedoeling was een universele manipulator te ontwikkelen, die zowel ingezet kan worden als hulpje voor zwaar lichamelijk gehandicapten als bij productieprocessen. De ervaringen die werden opgedaan bij deze ontwikkeling staan ter beschikking van de Nederlandse bedrijfswereld. Naast de elektronische, electromechanische en fijnmechanische aspecten van dit ontwerp, was een van de belangrijkste onderzoekterreinen het ontwikkelen van software en regelsystemen, die de bewegingen van de arm op de best mogelijke manier kunnen besturen.
“Manus” bestaat, zie figuur 3/17.2-1, uit een aantal segmenten, die door middel van gewrichten ten opzichte van elkaar

LEES OOK:
Hoofdstuk 3/6.4
Hoofdstuk 3/6.5
Hoofdstuk 3/17.1
Hoofdstuk 3/17.3



17.2 Voorbeelden van regelsystemen met vage logica

kunnen draaien. Deze gewrichten worden aangedreven door kleine elektromotoren, die zijn opgesteld in de basiskolom van de arm. Van daar uit worden de bewegingen via tandwielen, riemen en flexibele assen overgebracht naar de mechanische gewrichten.
De robot-arm heeft in totaal zes beweegbare gewrichten. In figuur 3/17.2-1 is aangegeven in welke richting deze gewrichten kunnen bewegen.



Figuur 3/17.2-1: De robot-arm "Manus", die door TNO werd ontwikkeld om ervaringen op te doen met besturingssoftware.

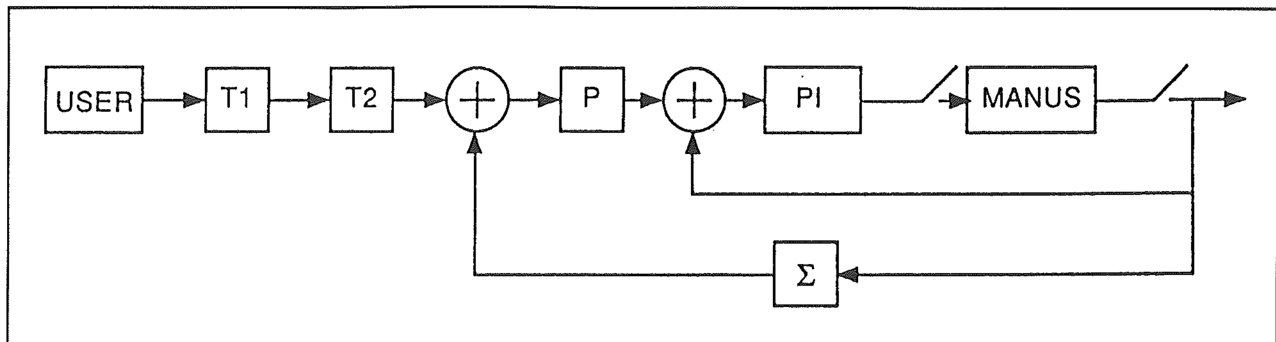
Bewegingsanalyse

Een van de grootste problemen bij het ontwerpen van een besturingssysteem voor een robot-arm is het gegeven, dat de beweging van een laag gelegen gewricht gevolgen heeft voor alle hoger gelegen gewrichten. Als men bijvoorbeeld gewricht A3 aanstuurt, zal het bovenste gedeelte van de arm gaan bewegen, waardoor natuurlijk ook de gewrichten A4, A5 en A6 mee gaan bewegen. Dit kan zeer ernstige gevolgen hebben voor de handeling, waarmee de robot bezig is. In de meeste gevallen zal de aansturing van één gewricht tot gevolg hebben dat ook de aansturing van de hoger gelegen gewrichten onmiddellijk bijgesteld moet worden. Neem als voorbeeld de menselijke handeling "het optillen van een kopje koffie". Men manipuleert de handgewrichten in eerste instantie dusdanig, dat men het kopje koffie kan optillen met horizontale stand van de vloeistof in de kop. Zou men nu het schoudergewricht bewegen, zonder onmiddellijk de polsgewrichten bij te sturen, dan is de kans groot dat men de inhoud van het kopje over het bureau zou uitgieten! Onze hersenen werken zo verfijnd (fuzzy?) dat dergelijke gecompliceerde bewegingen zonder nadenken worden uitgevoerd.
Robot-armen zijn echter dom en van iedere beweging moeten alle mogelijke consequenties voor de andere gewrichten razend snel door gerekend en eventueel gecorrigeerd worden.

Besturing

"Manus" wordt bestuurd volgens een Cartesiaans assenstelsel. Dat wil zeggen dat de menselijke bestuurder naar de grijper kijkt en door middel van zes knoppen de positie in de drie ruimtelijke richtingen X, Y en Z kan besturen. Het is nu de taak van de besturingssoftware om de gewenste verplaatsing van de grijper om te rekenen naar besturingscommando's voor de zes gewrichten.

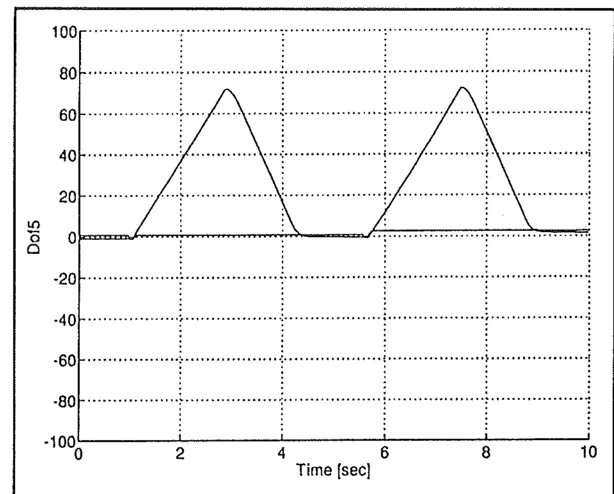
17.2 Voorbeelden van regelsystemen met vage logica



Figuur 3/17.2-2: Het "klassieke" regelsysteem voor een motor van "Manus".

Besturingssysteem

In eerste instantie werd gewerkt met een hard besturingssysteem, waarvan de structuur in figuur 3/17.2-2 is geschetst. De menselijke bestuurder "USER" voert via de knoppen de nieuwe coördinaten van de grijper in. In het logica-blok T1 worden deze X-, Y- en Z-waarden getransformeerd naar bewegingscoördinaten voor de gewrichten. Nadien moeten deze in het logica-blok T2 worden omgerekend naar bewegingscommando's voor de motoren. Natuurlijk is het onmogelijk om de motoren opeens op maximale snelheid te laten draaien. De constructie van de arm zou snel beschadigd raken en bovendien zou, in het voorbeeld van het kopje koffie, iedere bruuske beweging tot gevolg hebben dat de inhoud over de rand vliegt. Vandaar dat de snelheid van een motor wordt geregeld via een proportionele/integrerende regelaar PI. Het integrerende deel van deze regelaar zorgt ervoor dat de snelheid van de motor met een bepaalde vertraging naar de door T2 opgelegde waarde gaat. Voor het in de gaten houden van de positie van de as van de motor voldoet een proportionele regelaar P uitstekend. Natuurlijk zijn een aantal terugkoppelingen noodzakelijk die signalen, geleverd door sensoren, terugkoppelen naar het regelsysteem.

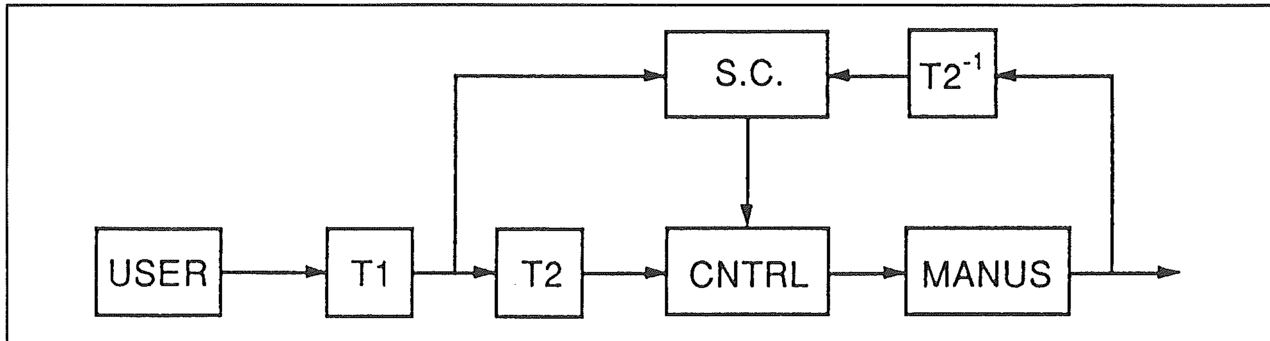


Figuur 3/17.2-3: Een voorbeeld van de optredende grote fouten bij de klassieke regeling.

Grote fouten

Uitgevoerd met de traditionele regelsystemen en harde beslissingsregels uit de "klassieke" regeltheorie bleek dat "Manus" bij sommige gecompliceerde bewegingen zeer grote fouten maakte. Deze waren niet te minimaliseren, waardoor "Manus" volstrekt onbruikbaar werd voor bepaalde toepassingen. In figuur 3/17.2-3 wordt als voorbeeld de foutcurve getekend die in een van de gewrichten optrad bij een bepaalde beweging.

17.2 Voorbeelden van regelsystemen met vage logica



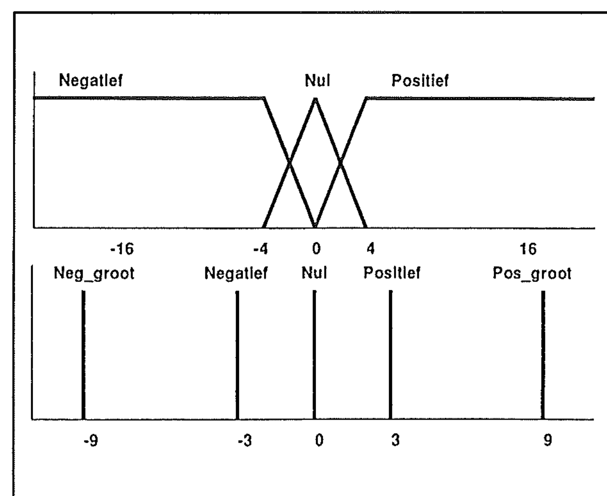
Figuur 3/17.2-4: Het invoegen van vage logica in het regelsysteem van de motoren van "Manus".

Bij deze beweging, die in totaal tien seconde in beslag neemt, treedt in gewricht 5 twee maal een afwijking van de gewenste stand af van niet minder dan 70°! Uit de grafiek blijkt duidelijk dat het regelsysteem weliswaar in staat is deze afwijkingen weer naar 0 te corrigeren, zodat de uiteindelijke stand van de grijper wel overeen komt met de gewenste stand. Maar het zal duidelijk zijn dat het absoluut niet mag voorkomen dat gedurende een gecombineerde beweging een van de gewrichten zo ver "doorslaat".

Invoegen van vage logica

Om deze problemen op te lossen werd een beroep gedaan op vage logica. In het regelsysteem werd een zogenoemde "Supervisory Control Module" ingebouwd. Deze supervisor bestaat uit een fuzzy logic systeem, waarin een kennisbank aanwezig is over de in de praktijk vastgestelde positioneringsfouten. Hoe deze vage logica in het systeem werd geïntegreerd is te zien in figuur 3/17.2-4. De vage logica module SC heeft twee ingangsgrootheden, "e_joint" en "de_joint", die absolute en relatieve fouten in de positie van de gewrichten meten en een uitgangsgrootheid "v_add", die een extra stuursignaal genereert voor het besturen van de snelheid van de motor.

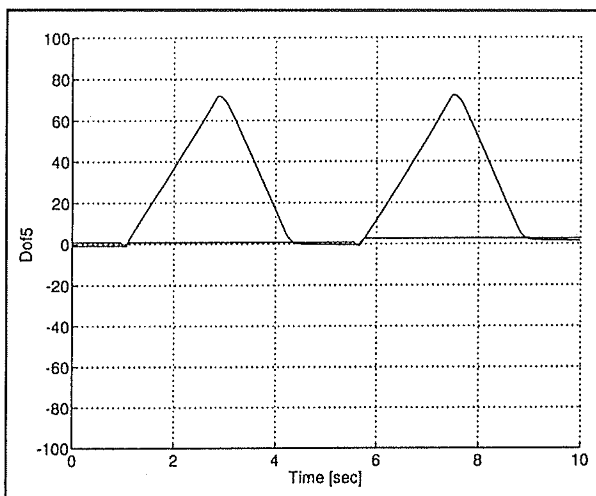
De lidmaatschapsfuncties voor deze drie grootheden, die naar veel experimenteren werden geoptimaliseerd, zijn voorgesteld in figuur 3/17.2-5. De bovenste lidmaatschapsfunctie geldt voor beide ingangsgrootheden en kan ontleed worden in de drie termen "Negatief", "Nul" en "Positief". Deze termen hebben natuurlijk betrekking op de vastgestelde afwijking in de positie. De onderste lidmaatschapsfunctie geldt voor de uitgangsgrootheid "v_add" en bestaat uit vijf singletontermen.



Figuur 3/17.2-5: De lidmaatschapsfuncties voor de in- en uitgangsgrootheden.

17.2 Voorbeelden van regelsystemen met vage logica

Uit figuur 3/17.2-5 blijkt dat de lidmaatschapsfuncties voor zelfs vrij gecompileerde regelingen in de praktijk erg eenvoudig kunnen zijn! Er bestond in dit voorbeeld geen enkele behoefte om ingewikkelde S-, PI- of Z-functies op te nemen of gebruik te maken van powered hedges.
Omdat er in totaal zes ingangstermen zijn kunnen er negen beslissingsregels opgesteld worden. Deze hebben de vorm van:
ALS e_joint IS negatief EN de_joint IS negatief DAN v_add IS groot
Voor de defuzzificatie wordt gebruik gemaakt van de mean of maximum strategie MOM. Hierbij wordt het gewogen gemiddelde van de actieve beslissingsregels bepaald en hiermee het extra stuursignaal voor de motor gegenereerd.



Figuur 3/17.2-6: Dezelfde foutkarakteristiek als in figuur 3/17.2-3, maar nu nadat er vage logica in de besturing van de robot-arm werd ingebracht.

Resultaat

Het resultaat van het invoeren van vage logica in het besturingssysteem van "Manus" is spectaculair! In figuur 3/17.2-6 is de foutgrafiek gegeven van dezelfde motor A5, die ook in figuur 3/17.2-3 als voorbeeld werd genomen. Dank zij vage logica is de maximale afwijking in de positie van dit gewricht terug gebracht tot +/-2°!

Het roer van een schip

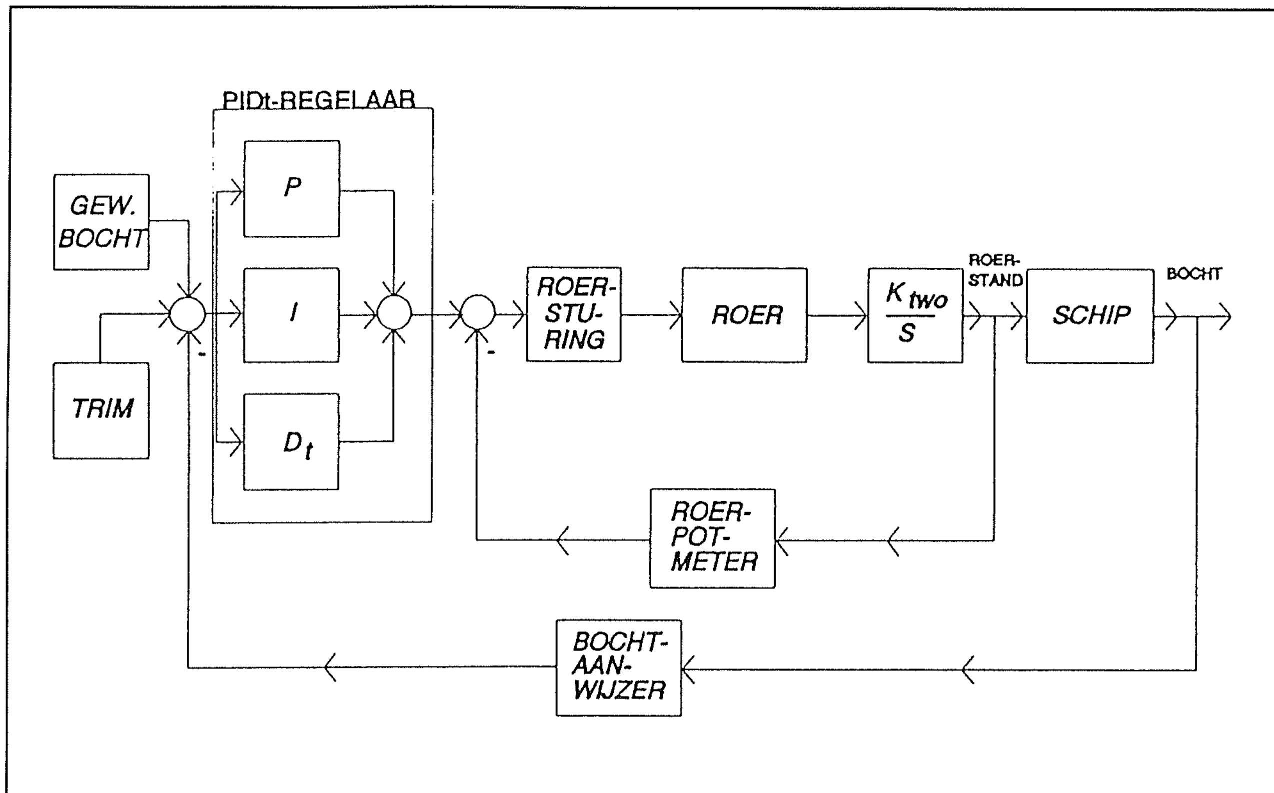
Inleiding

De meeste moderne schepen hebben een automatische piloot oftewel stuurautomaat. Deze laat het schip de door de roerganger uitgezette koers volgen. Dat lijkt eenvoudiger dan het is. Op een varend schip werken een heleboel krachten in, bijvoorbeeld de oppervlaktestromingen in het zeewater, de kracht van de wind, maar ook door de leek onvermoede factoren, zoals de kracht die een niet goed verdeelde lading uitoefent.
Volgens de regels van de klassieke regeltheorie kan een dergelijk probleem het best opgelost worden door gebruik te maken van een zogenoemde PID-regelaar. PID is de afkorting van "proportioneel, integrerend en differentiërend". Het zou te ver gaan om in dit verband toe te lichten waar deze begrippen voor staan. Het komt er in het kort op neer dat een dergelijk regelsysteem zowel compenseert voor afwijkingen op korte termijn (differentiërend) als voor afwijkingen op lange termijn (integrerend).

Het traditioneel regelsysteem

Het traditionele regelsysteem is voorgesteld in figuur 3/17.2-7. Als ingangsgrootheden worden "GEW. BOCHT" en "TRIM" gebruikt. "TRIM" bestaat in feite uit een heleboel instelbare tijdconstanten, die de eigenschappen van een individueel schip aanpassen aan het standaard regelsysteem.

17.2 Voorbeelden van regelsystemen met vage logica



Figuur 3/17.2-7: Het klassieke besturingssysteem van het roer van een schip.

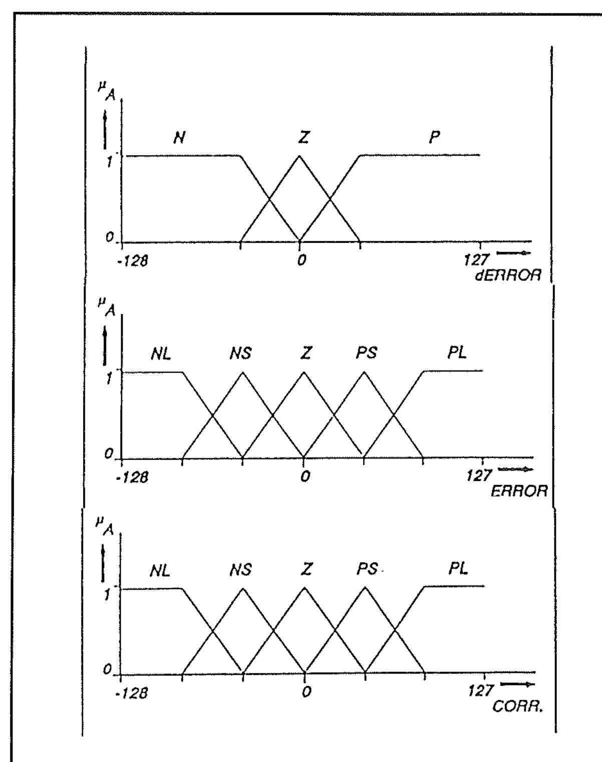
Het regelblok K_{two}/S staat voor de overbrengingsverhouding tussen de besturing en het eigenlijke roer van het schip.
Nu is echter geen enkel schip gelijk aan een ander en de vaareigenschappen van een bepaald schip kunnen zelfs van reis tot reis erg verschillen, als gevolg van de eigenschappen van de lading. Het zal duidelijk zijn dat dergelijke individuele eigenschappen erg moeilijk in een hard regelsysteem zijn te vangen. Een typisch klusje voor fuzzy logic, waar de kennisbank van het individueel schip goed van pas komt om geïntegreerd te worden in een vaag systeem!

De vage oplossing
Voor het besturen van het roer van een schip door middel van vage logica moet men natuurlijk weer eerst in- en uitgangsvariabelen gaan definiëren. In dit geval zijn dat ook weer twee foutsignalen, namelijk "FOUT" en "dFOUT". De eerste variabele wordt afgeleid uit het regelsysteem zélf, de tweede komt uit een kennisbank, waarin de specifieke eigenschappen van het betreffende schip zijn opgeslagen. Als uitgangsvariabele levert het vage proces een correctiesignaal "CORR", waarmee een corrigerend signaal naar het roer wordt gestuurd.
Uit praktijkervaring blijkt dat men de lidmaatschapsfuncties van deze drie variabelen in termen moet splitsen volgens het schema van figuur 3/17.2-8. Ook nu zijn zuiver lineaire termen voldoende voor het beschrijven van het gehele proces. De variabele "ERROR" wordt ingedeeld in vijf termen, van NL (Negative Large) tot PL (Positive Large). De variabele "dERROR" wordt ingedeeld in drie termen, N(egative), Z(ero) en P(ositive). De uitgangsva-

17.2 Voorbeelden van regelsystemen met vage logica

riabele "CORR" bevat ook vijf termen, van NL (Negative Large) tot PL (Positive Large).
De beslissingsregels voor het systeem zijn al even eenvoudig en hebben de vorm van:
ALS ERROR IS L EN dERROR IS H DAN CORR IS H
en:
ALS ERROR IS M OF dERROR IS M DAN CORR IS M
In figuur 3/17.2-9 is het volledige inferentie-proces voor deze set van twee beslissingregels samengevat. Duidelijk blijkt in de bovenste figuur de invloed van de fuzzy-operator EN op het geheel. Alleen de kleinste waarde van de twee ingangsvariabelen wordt nu doorgekoppeld naar de uitgang. In de onderste tekening blijkt de invloed van de OF-operator. Nu wordt de hoogste ingangswaarde doorgekoppeld naar de uitgang. Uit deze figuur blijkt dat in dit voorbeeld gekozen is voor een inferentie volgens de MAX-PROD strategie en dat de defuzzificatie volgens de zwaartepunt-methode COG plaats vindt.

Conclusie
Ook nu blijkt dat een in wezen zeer ingewikkeld proces als het automatisch besturen van een schip met vage logica heel erg eenvoudig op te lossen is. Zoals steeds is het moeilijkste probleem het opstellen van de kennisbank. Hiervoor is vrij weinig theoretische kennis vereist, maar des te meer praktijkervaring van de mensen die het schip en zijn gedragingen onder alle omstandigheden kennen. De lidmaatschapsfuncties zijn erg eenvoudig en ook het inferentie-mechanisme is nadien snel op te stellen. Ook bij dit voorbeeld blijkt dat de regelnauwkeurigheid van het systeem sterk toeneemt door het invoeren van vage logica. Ter illustratie is in figuur 3/17.2-10 het verloop van de positie van het schip in functie van de tijd weergegeven. Hieruit blijkt (bovenste grafiek) dat het schip rustig reageert en na ongeveer een halve minuut precies op de voorgeschreven koers zit, zonder dat er sprake is van enige afwijking.

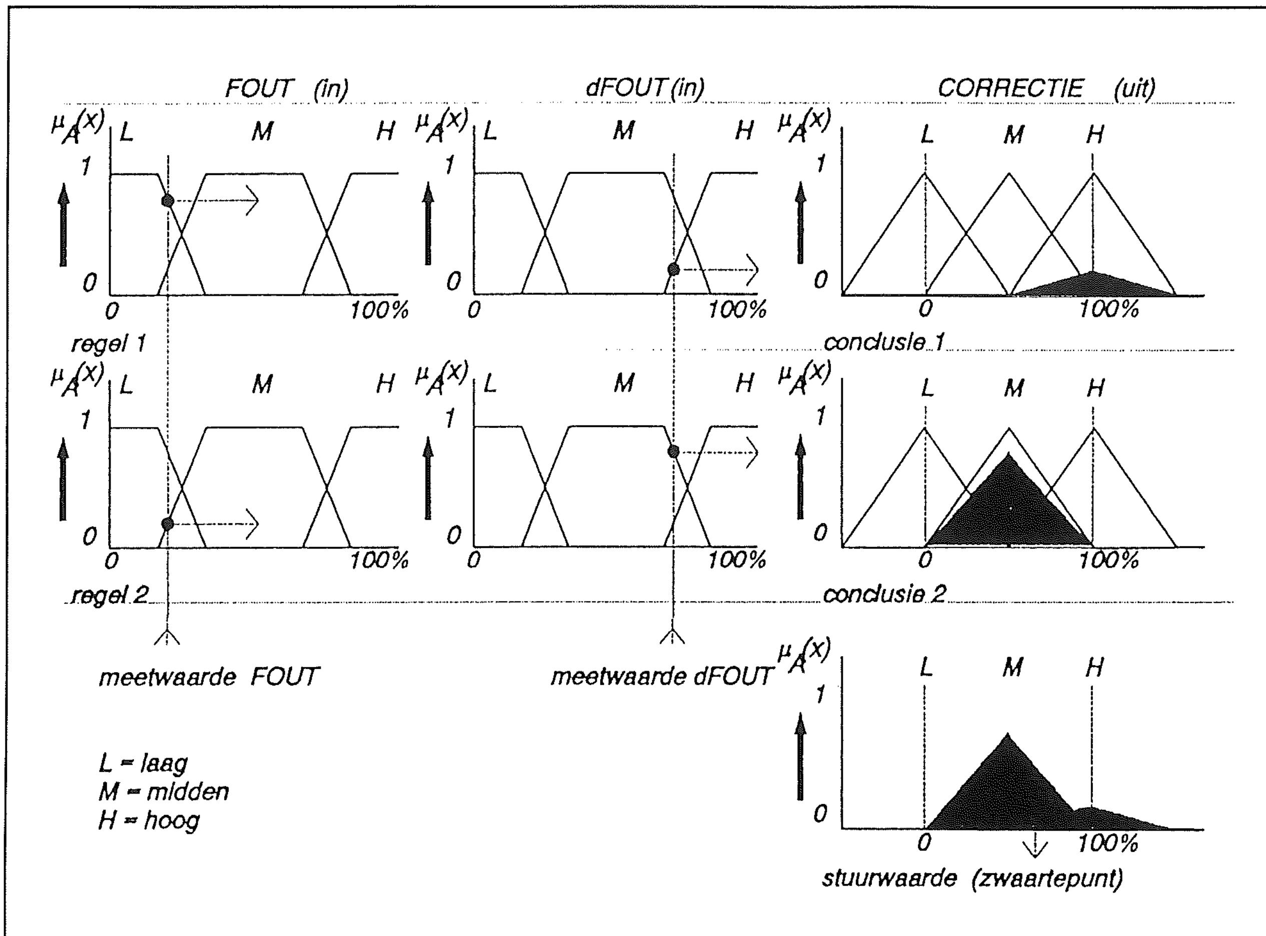


Figuur 3/17.2-8: De drie lidmaatschapsfuncties van de twee ingangsvariabelen en de ene uitgangsvariabele.

Kwaliteitsbewaking

Inleiding
Bij de productie van onderdelen uit rubber is het van groot belang dat de kwaliteit van de basisgrondstof constant blijft. Deze grondstof bestaat uit een menging van natuurlijk en synthetisch rubber, olieën en andere chemicaliën.

17.2 Voorbeelden van regelsystemen met vage logica



Figuur 3/17.2-9: Het inferentie-proces bij het voorbeeld van het aansturen van het roer van een schip.

Monsters van dit mengproduct worden onder de microscoop bekeken en vergeleken met de zogenoemde "Philips-schaal". Dit is een industriële standaard, die tien kwaliteitsklassen van het basismateriaal definieert, aan de hand van goede menging, grootte van de te mengen deeltjes, etc. Het beoordelen van de kwaliteit aan de hand van deze schaal gebeurt in de meeste gevallen nog steeds door mensen, die monster na monster met een microscoop onderzoeken en vergelijken met de tien referentiebeelden van de Philips-schaal. Om een idee te geven hoe moeizaam dit gaat is in figuur 3/17.2-11 het standaard-patroon getekend van vier van de tien kwaliteitsklassen van deze schaal.

Bij deze visuele beoordeling van de monsters komt een heleboel ervaring kijken en het proces is bovendien zeer subjectief, omdat de overgangen tussen de tien kwaliteitsklassen niet erg goed gedefinieerd kunnen worden. Bovendien in het proces zeer arbeidsintensief, iedere test duurt ongeveer vijf minuten, waarna er nog handmatig een kwaliteitsverslag moet opgesteld worden en een foto gemaakt voor het archief van de afdeling kwaliteitsbewaking.

Een fuzzy systeem

Natuurlijk bestaat de behoefte om dit arbeidsintensieve proces te automatiseren. Door het Duitse "Fuzzy Demonstration

17.2 Voorbeelden van regelsystemen met vage logica

Centrum" uit Dortmund werd, in samenwerking met enige grote Duitse rubberverwerkende industrieën, een geautomatiseerd testsysteem ontwikkeld. Het zal wel duidelijk zijn dat vanwege het zeer subjectieve karakter van de beoordelingen er geen sprake kan van zijn dit proces aan harde logica over te laten. Vandaar dat met succes een poging werd ondernomen om dit proces te fuzzyficeren.

Meetopstelling

De kwaliteitscontrole gaat uit van monstertjes van 6 bij 6 bij 6 mm^3. Deze worden door middel van een CCD TV-camera, bevestigd op een microscoop, omgezet in een beeld van de oppervlaktestructuur en via een normale digitiser-kaart in een PC ingelezen.

De beelden worden bewaard in het in de grafische wereld vrijwel gestandaardiseerde TIF-formaat. Vervolgens wordt er speciale beeldherkenningssoftware op de beelden losgelaten, die ieder monster onderzoekt op grootte van de korrels, aantal van de korrels, grijswaarde van alle pixels, etc. Deze gegevens dienen als ingangsvariabelen voor een fuzzy logic beslissingssysteem. In de kennisbank van dit systeem zijn de tien referentiebeelden van de Philips-schaal opgeslagen.

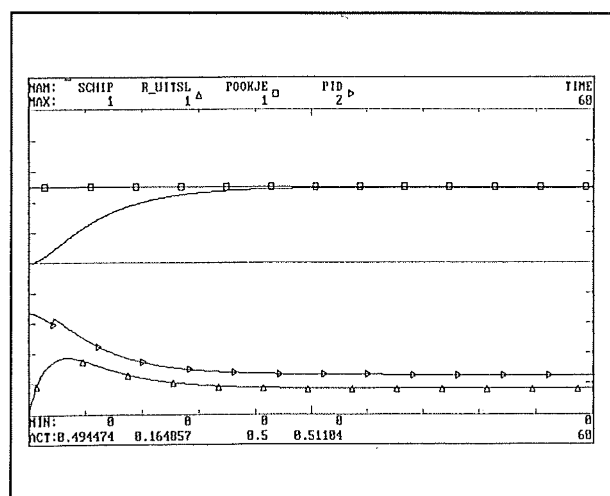
Het klassificeren van een monster duurt met het fuzzy gecontroleerde systeem slechts enkele seconden en blijkt in de praktijk veel betrouwbaarder te werken dan een klassificering door mensen.

Hoe de fuzzificatie en defuzzificatie precies in zijn werk gaat wordt op dit moment nog niet geopenbaard, omdat het systeem nog gepatenteerd moet worden.

Conclusie

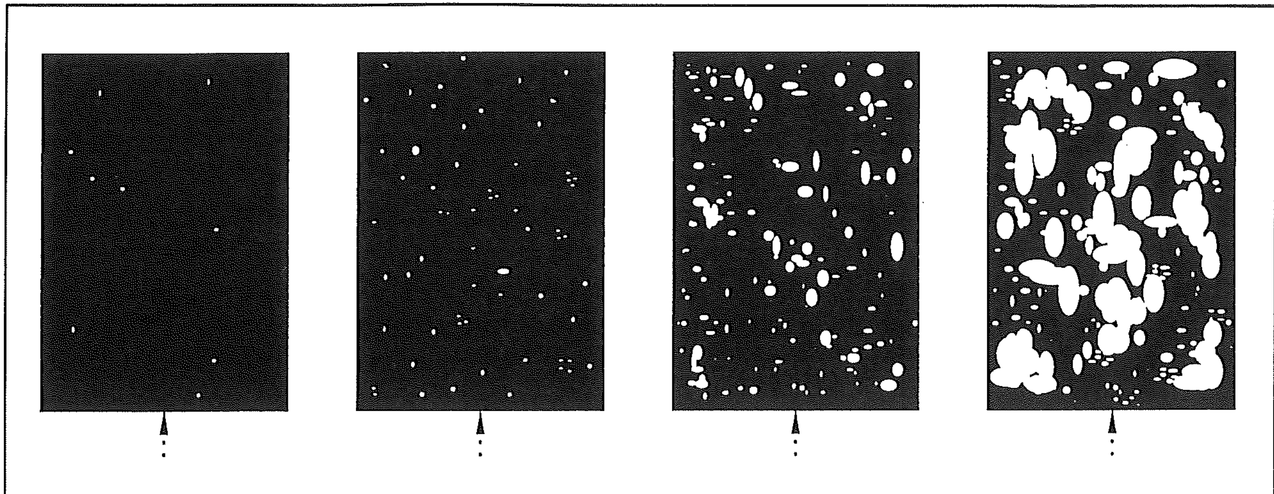
Ook uit dit voorbeeld blijkt duidelijk dat vage logica niet alleen grote tijdwinst kan opleveren, maar bovendien een aanmerkelijke verbetering van de kwaliteit van een regelsysteem tot gevolg heeft. Bovendien geeft dit voorbeeld een fraaie blik op de toekomst.

Binnen vijf jaar zal gefuzzificeerde beeldherkenning op tal van gebieden standaard zijn en kan men misschien de huissleutel vervangen door een TV-camera, aangesloten op een fuzzy-systeem dat het opgenomen beeld van de persoon voor de deur vergelijkt met een referentiebibliotheek en alleen de deur opent als het opgenomen beeld overeen komt met een van de beelden uit de bibliotheek. Toekomstmuziek? Zeer zeker niet! Want reeds nu is er een systeem op de markt, waarbij met fuzzy logic beoordeelde TV-beelden van een menselijk hoofd worden gebruikt voor het opnemen van geld via een betaalautomaat.



Figuur 3/17.2-10: Respons van het fuzzy-gestuurde schip op een scherp stuurcommando.

17.2 Voorbeelden van regelsystemen met vage logica



Figuur 3/17.2-11: Vier kwaliteitsklassen uit de Philips-schaal, waarmee men de kwaliteit van de basisgrondstof voor het maken van rubber onderdelen kan beoordelen.

3/17.3

Principes van fuzzy-processoren

Inleiding

Nieuwe denkwijzen noodzakelijk

Fuzzy logic werkt, dat zal wel duidelijk zijn geworden uit de vorige hoofdstukken, op een heel andere manier dan binaire logica. De binaire logica is ontwikkeld met als basis de harde logica van binaire elektronica. Vandaar dat deze logica zo gemakkelijk op een heel eenvoudige manier in elektronische schakelingen vertaald kan worden. Het ontstaan van de vage logica ligt echter in de theoretische wiskunde en heeft zich van daaruit ontwikkeld naar praktische toepassingen. De vraag kan dan ook gesteld worden hoe het mogelijk is vage regels in harde elektronica uit te voeren. Nu, dat kan niet zonder meer en hoewel er speciale fuzzy-processoren en -coprocessoren bestaan werken deze natuurlijk nog altijd "hard". Dat ligt nu eenmaal in de aard van het elektronische beestje!

In het algemeen komt het er op neer dat ingangsgrootheden in ieder geval vertaald moeten worden naar harde binaire waarden. In de speciale fuzzy-processoren zijn ***harde*** algoritmen (rekenprocedures) ingebouwd, die de ***zachte*** beslissingsregels kunnen uitrekenen. De beslissingsregels en alle andere regels van de inferentie en de defuzzyficatie zijn ofwel standaard in een ROM aanwezig, of kunnen met geschikte software in een geheugen ingelezen worden.

Samenwerking zacht/hard

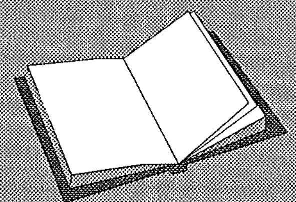
Het zal duidelijk zijn dat fuzzy-processoren gewoon hard werken. In de meeste gevallen moeten zij samenwerken met een microcontroller uit de harde logica. Alleen het berekenen van de specifieke in de processor aanwezige fuzzy-algoritmen wordt dan aan de fuzzy-processor overgelaten, al de rest van het rekenwerk komt voor rekening van de binaire processor.

Twee voorbeelden

Om een indruk te geven van de manier waarop fuzzy-processoren werken, worden twee van dergelijke processoren besproken:

- de SAE81C99 "fuzzy coprocessor" van Siemens;
- de VY86C500 "fuzzy acellerator" van VLSI Technology.

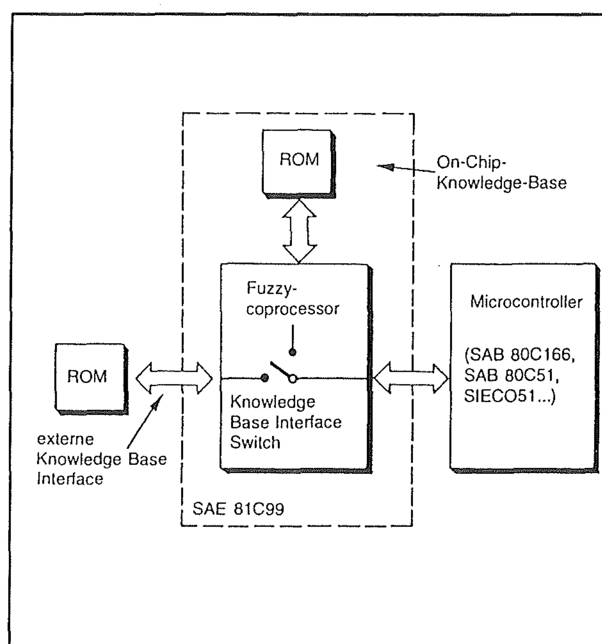
LEES OOK:
Hoofdstuk 3/17.1
Hoofdstuk 3/17.2



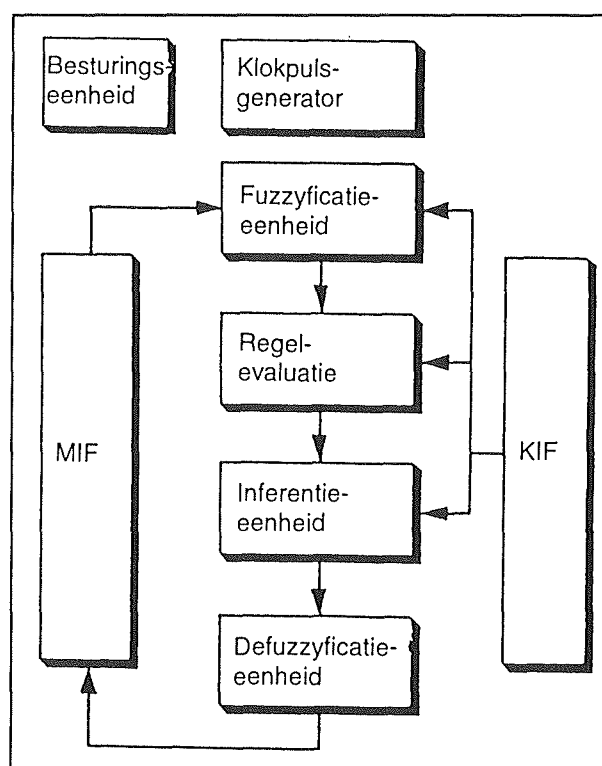
17.3 Principes van fuzzy-processoren

De SAE81C99 van Siemens

Coprocessor
De SAE81C99 is een fuzzy-coprocessor, hetgeen betekent dat deze schakeling niet zelfstandig kan werken, maar ontwikkeld werd als uitbreiding op een normale microcontroller van de door Siemens ontwikkelde SAB80xxx-reeks.
Het blokschema van het systeem is voorgesteld in figuur 3/17.3-1. Veel gebruikte standaard beslissingsregels van het beslissingsproces en veel voorkomende lidmaatschapsfuncties zijn ingeprogrammeerd in een ROM. Door middel van de zogenoemde "Knowledge Base Interface Switch" kan men echter ook omschakelen naar een extern geheugen, waar voor de toepassing specifieke lidmaatschapsfuncties en beslissingsregels in opgeborgen kunnen worden. De coprocessor bezit acht verschillende algoritmen voor het doorlopen van het volledige linguïstische protocol.
Het werkingsprincipe is als volgt:
- eerst selecteert de microcontroller de gewenste kennisbank;
- nadien bestuurt de microcontroller alle analoge ingangswaarden en geeft deze door aan de fuzzy-coprocessor;
- de fuzzy-coprocessor doorloopt nu het volledige fuzzy-proces en berekent de harde waarden van de uitgangsgrootheden;
- de fuzzy-coprocessor geeft een interruptsignaal af, waardoor de controller weet dat de berekening is afgesloten;
- de berekende uitgangswaarden worden opgehaald in het data-register van de coprocessor en gebruikt voor het besturen van ingangsgrootheden.

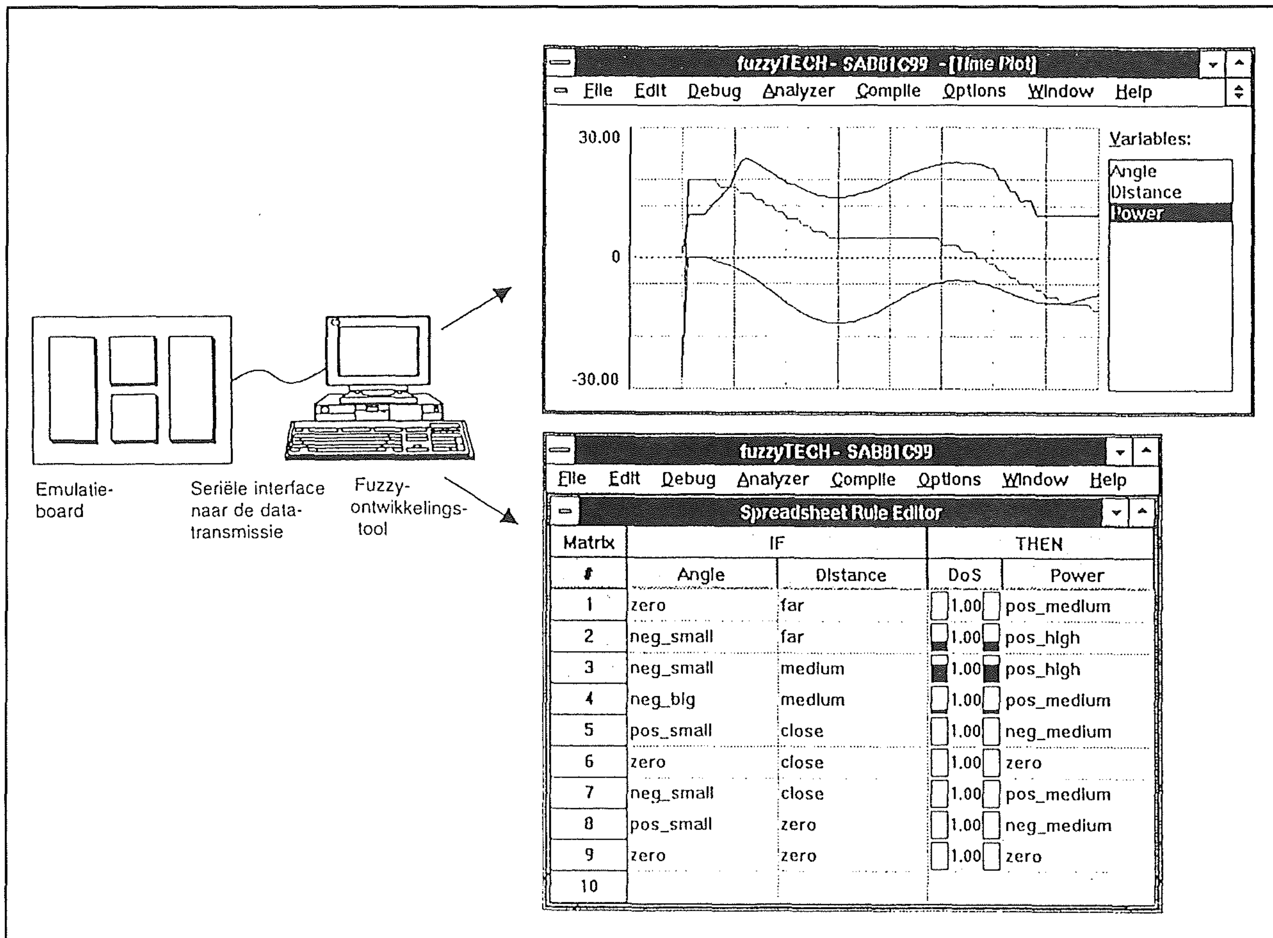


Figuur 3/17.3-1: De samenwerking tussen de fuzzy-coprocessor en een normale controller.



Figuur 3/17.3-2: Het blokschema van de SAE81C99.

17.3 Principes van fuzzy-processoren



Figuur 3/17.3-3: Het software-pakket "Fuzzy-TECH" wordt gebruikt voor het programmeren van fuzzy-systemen in de SAE81C99.

Blokschema van de SAE81C99

In figuur 3/17.3-2 is het blokschema van deze Siemens-schakeling getekend. De ingangswaarden worden via de "Microprocessor InterFace" (MIF) naar de fuzzyficatie-eenheid doorgekoppeld. De "Knowledge Base Interface" (KIF) zorgt ervoor dat de berekende waarden op de juiste plaats terecht komen in alle opgestelde beslissingsregels. Nadien worden de ingevulde regels verder verwerkt in de "Regelevaluatie", de "Inferentie-eenheid" en de "Defuzzyficatie-eenheid". In deze drie blokken kunnen telkens twee verschillende verwerkingsmogelijkheden worden ingesteld. Uiteraard kan men hierbij de standaarden Min-Max en het Center of Gravity algoritme selecteren.

De SAE81C99 werkt op een interne klok van 20 MHz en met 8 bit brede gegevens. De coprocessor is in staat in ongeveer 80 µs een algoritme uit te voeren dat bestaat uit:

- acht ingangsgrootheden;
- zeven termen per lidmaatschapsfunctie;
- één uitgangsgrootheid met een lidmaatschapsfunctie met maximaal acht termen;
- 256 beslissingsregels.

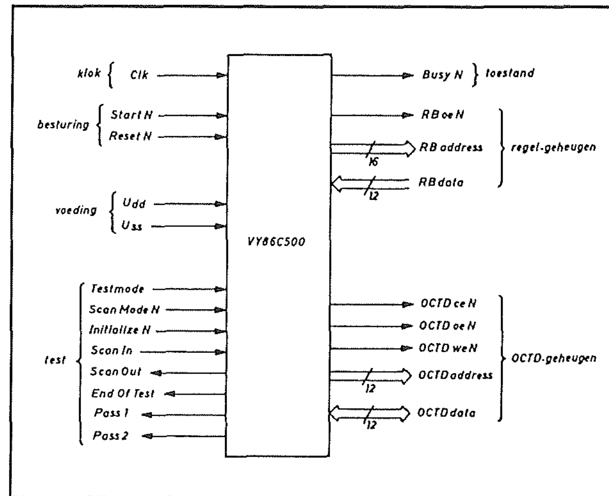
Software

Uiteraard wordt het ontwerpen van fuzzy-systemen ook hier vergemakkelijkt door het beschikbaar stellen van een software ontwikkelingssysteem. Dit systeem, "Fuzzy-TECH" genoemd, kan draaien op een normale PC, die via een normale seriële interface communiceert met de hardere componenten van het ontwikkelingssysteem. Met deze software, zie figuur 3/17.3-3, is het mogelijk de beslissingsregels en lidmaatschapsfuncties op een interactieve manier te wijzigen.

De VY86C500 van VLSI Technology

Blokschema

De VY86C500 wordt door de fabrikant een "fuzzy-acellerator" genoemd. Ook deze schakeling is een coprocessor, want hij moet samenwerken met een normale binaire processor die de besturing van het fuzzy-systeem voor zijn rekening neemt.
Het IC werkt op een kloksnelheid van 20 MHz en heeft een verwerkingsbreedte van 12 bit. Volledige fuzzy-algoritmen worden in tientallen µs tot tientallen ms uitgevoerd, afhankelijk van de ingewikkeldheid van het probleem. Het blokschema van deze fuzzy-processor is getekend in figuur 3/17.3-4. Twee geheugen-interfaces domineren de aansluitingen. Het zogenoemde "OCTD-geheugen" kan 4.096 woorden van 12 bit bevatten en wordt gebruikt voor het opbergen van tussenresultaten, het bewaren van waarden van ingangsgrootheden en het uitvoeren van de berekende waarden van de uitgangsgrootheden. Het tweede geheugen is het "Regelgeheugen".



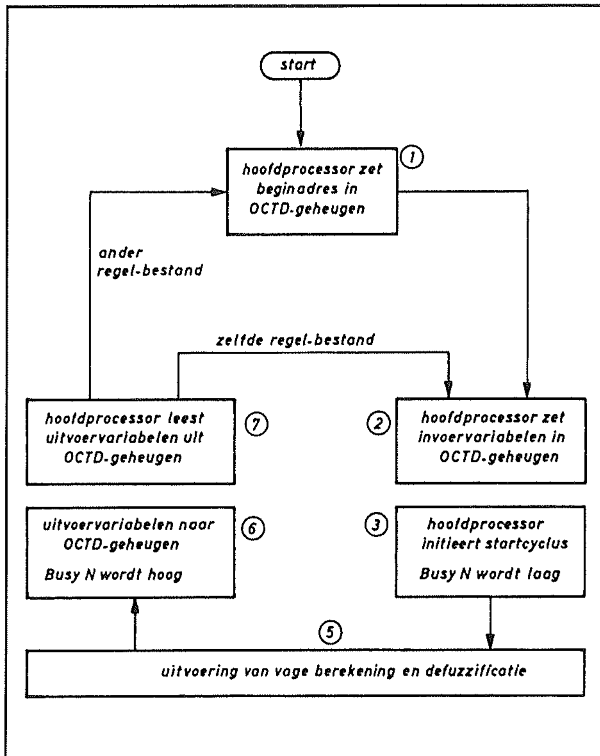
Figuur 3/17.3-4: Het blokschema van de vage processor VY86C500.

De 16 bit brede adresbus van dit geheugen laat toe maximaal 65.536 12 bit woorden op te bergen. In dit geheugen kan men alle beslissingsregels opslaan.

Werking

De procedure voor het initiëren en afhandelen van een vage bewerkingscyclus wordt voorgesteld in figuur 3/17.3-5. De hoofdprocessor zet eerst het startadres van het beslissingsregelbestand in het OCTD-geheugen. Nadien moet de hoofdprocessor alle ingangsvariabelen onder digitale vorm in hetzelfde geheugen inlezen. Het verwerken van de gegevens door de fuzzy acellerator wordt ingeleid door het BusyN-signaal "L" te maken. Vervolgens wordt het proces volledig overgenomen door de VY86C500. Er zijn nogal wat parameters die van invloed zijn op de totale tijdsduur van het fuzzy-proces. Hierbij speelt het aantal beslissingsregels uiteraard een voorname rol. De VY86C500 kan zowel lineaire termen als S-functies verwerken. Het laatste soort gebruikt, vreemd genoeg, veel minder geheugenruimte maar het berekenen ervan neemt veel meer tijd in beslag.

17.3 Principes van fuzzy-processoren



Figuur 3/17.3-5: De werkingsprocedure van de fuzzy acellerator VY86C500.

Nadat de fuzzy acellerator klaar is met het verwerken van alle regels en de inferentie en de defuzzyficatie zijn uitgevoerd, worden de berekende waarden van de uitgangsgrootheden beschikbaar gesteld aan de hoofdprocessor. Het signaal BusyN wordt weer "H" gestuurd. De hoofdprocessor kan nu de berekende waarden van de uitgangsvariabelen uit het OCTD-geheugen uitlezen.

Prestaties

De VY86C500 kan maximaal 850.000 beslissingsregels per seconde evalueren. Uiteraard hangt de totale verwerkingsduur van een heleboel gegevens af.

- **Voorbeeld 1:**
  - twee ingangen;
  - vijf lineaire lidmaatschapsfuncties van 6 termen;
  - twaalf beslissingsregels;
  - één uitgangsgrootheid.
- Verwerkingsduur: 22 µs
- **Voorbeeld 2:**
  - dertig ingangsgrootheden;
  - zeven lidmaatschapsfuncties met zes termen;
  - duizend beslissingsregels;
  - vier uitgangsgrootheden.
- Verwerkingsduur: 3,6 ms

17.3 Principes van fuzzy-processoren